SPARK

Reports Builder for SharePoint

User Manual

Ver. 2.0.0.0

SPARKnit Inc.
11420 Delores Ferguson Ln
Charlotte, NC 28277
Tel: +1 (704) 559-5950
info@sparknit.com | www.sparknit.com

Table of Contents ...

1 The Report Designer Workspace

SPARK Reports designer allows you to create customized reports within your SharePoint environment quickly and easily, these reports will run where your users need them. You can run SPARK reports from anywhere and on any device, desktop, mobile devices, tablets, iPads and on their preferred web browser, and across operating systems.

The report designer workspace (canvas) contains six main areas:

- **Top Ribbon** at the top of the designer page.
- **Report Toolbox** on the left side of the page.
- **Controls Properties panes** on the right side of the page**.**
- **Reports Properties** on the right side of the page**.**
- **Report Design Workspace** on the center of the page.
- **Ad-Hoc Report's Form Design Workspace** on the center of the page.

1.1 Report Toolbox

The **Report Toolbox** displays the controls that the designer can add to the report in the design workspace (canvas).

Please refer to **Control Properties** in **Controls** section for more information on configuring a control.

In addition to the controls displayed in the Report Toolbox area, the system presents all Query columns selected in the **Query Builder** query design, so the user can simply drag/drop the desired column(s) on the **Report Table** control area.

1.1.1 Hiding / Showing Report Toolbox

- **To hide the Toolbox:** Click the Drawing Pin icon located on the top right corner of the toolbox header. The toolbox will collapse, leaving a visible tab.
- **To make the Toolbox remain visible:** Click the Drawing Pin icon to pin the panel open.

1.2 Adding Controls to the Design Workspace

To begin designing a report, add controls to the report design workspace and configure each control.

Drag a control from the **Toolbox** and **drop** it into position onto the Report/form design workspace.
In case of working on a Report Table control, drag the column from the Columns Section in the Toolbox and drop it directly on the control to add that column to the report's table and be able to configure its properties later on.

Note: Use the arrow keys to reposition a control once it is on the Report/form design workspace.

1.3 Selecting Controls

Select any control on the report design workspace by clicking with the mouse on it. The **Control Properties** will appear on the right side of the Report/form design workspace. By default, the properties pane of the report **Collapses** and the control properties pane appears, you can **Collapse or Expand** it by clicking on the +/− icons.

To select multiple controls:

- Hold down the **CTRL** button down and click with the mouse to select additional controls.
- Select multiple controls using the mouse, by clicking on an empty area, while dragging the mouse over the desired controls, the controls will appear selected.

1.4 Controls Properties

Please refer to **Control Properties** in Controls section for more information on configuring a control.

1.5 Report Properties

Please refer to Report Properties section for more information on configuring a report.

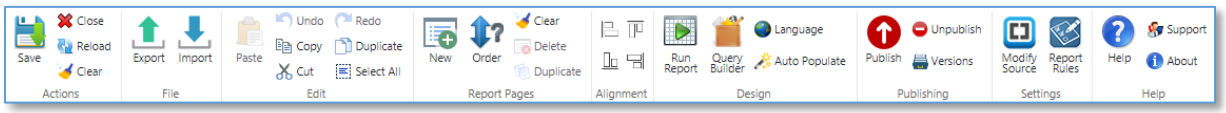
1.6 Rules Pane

You can use Rules to add dynamic formatting, formulas, actions or validations, which affect the controls behavior on the report at runtime.

Please refer to the Rules section for more information on the Rules Pane.

1.7 Report Design Top Ribbon

Located at the top of the designer's page. Top ribbon groups and related buttons descriptions are listed below.



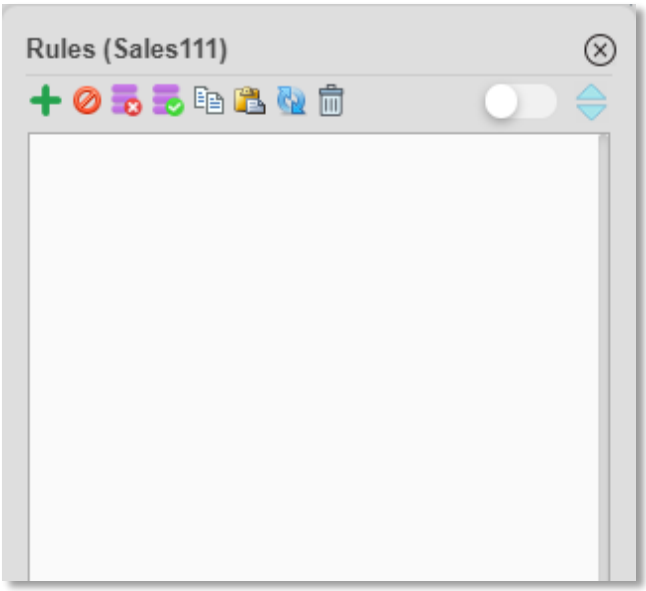
- **Actions Group:**
  - **Save:** Will save the current report design and generate a new version of the template.
  - **Close**: Clicking **Close** will close the report designer and return to the original location (source location). If the current report has not been saved yet, a warning dialog will appear to confirm saving before closing the page. If the user confirm closing without saving the report, the unsaved design will be lost.
  - **Reload**: Clicking **Reload** will reload/refresh the current report to the updated version of a saved form (if its save).
  - **Clear:** Clicking **Clear** will reset and remove all report's configurations, controls, Ad-Hoc form, rules, scripts and styles including Report Pages. Note that this will not affect the report query design and its configuration.
- **File Group:**
  - **Import Report**: Imports a SPARK report design file ".srb" to the current opened report designer including all rules, report design, Ad-Hoc form's design, controls, custom scripts, styles, report's query and its settings.

    Note: SPARK Reports Builder has a Checksum feature, allowing the designer to get a summary list of all issues that occur when importing an exported report especially if it was from another

SharePoint environment, this is very useful to trace and debugging imported reports.

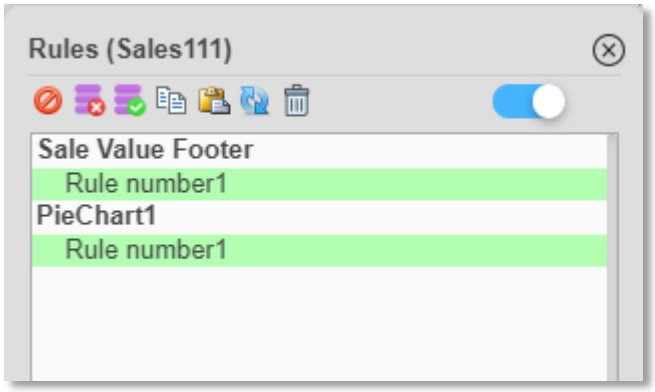
- **Export Report**: Exports the current report design, including all rules, report design, Ad-Hoc form's design, controls, custom scripts, styles, report's query and its settings to a file with extension ".srb" referring to (SPARK Reports Builder).

- **Edit Group:**
  - **Clipboard options (Cut, Copy and paste)**: Provides the option to **Cut** / **Copy** and **Paste** controls onto the current report page or to a different report's pages.
  - **Undo:** Undo the last action. Shortcut keys are available for the clipboard options (Ctrl + z). Note: The undo action will not affect changes in rules.
  - **Redo:** Reverses the most recent Undo action. Shortcut keys are available for the clipboard options (Ctrl + y).
  - **Duplicate:** Duplicates the selected control(s).
  - **Select All:** Select all control in the current report page. Shortcut keys are available (Ctrl + a).
- **Report Pages Group:**
  - **New:** Create a new report's page.
  - **Clear:** Clear all controls settings, configurations and design on the current opened report's page.
  - **Delete:** Delete the current **Report Page**. Note that the default built-in **Details Page** cannot be deleted.
  - **Duplicate:** Duplicate the current selected/opened report's page. A popup dialog will appear to specify the name of the new report's page.
- **Alignment Group:**
  - **Alignment:** Alignment tools use to Align (left, right, top bottom) the selected controls. Note: this tool only visible when two or more controls are selected.
- **Control Group:** This group appears when only one control is selected.
  - **Control Rules:**
    To open the rules panel, select the control you need to add rules to, you will see that the rules panel appears at the left side of the report design workspace.



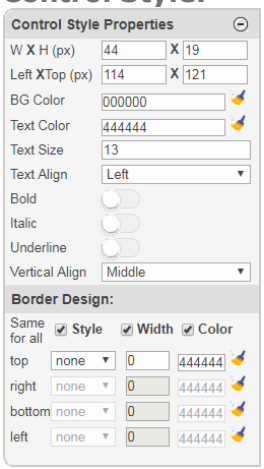
By default, the rules panel will show only rules that are associated to the selected control. The rules panel has the following buttons in order to work with rules associated with the selected control, these buttons are:

- **Show all:** By switching ON this toggle switch, you can view all related rules for the report and its controls. Double click on the rule will open the Rule Manager dialog to manage this rule the way you want.



- **Create:** Create a new rule for the selected control.
- **Disable/Enable:** Disable or enable the selected rule on the selected control. If the selected rule is disabled, the enable icon will be shown, If the selected rule is enabled, the disable icon will be shown. Note that disabled rules will not be executed at runtime.
- **Enable All:** Enable all rules on the selected control.
- **Disable All:** Disabled all rules on the selected control.
- **Copy Rule:** Copy the selected rule for the control (you need first to select a rule in order to enable this button).
- **Paste Rule:** Paste the copied rule from another control to the current/selected control.
- **Refresh:** Refresh rules list in the selected control.
- **Delete Rule:** Delete the selected rule (you need to select a rule in order to enable this button).
- **Up:** Move the selected rule one level up in the rules list; uppers rules have higher priority than the down ones (you need to select a rule in order to enable this button).
- **Down:** Move the selected rule one level down in the rules list; lower rules have lower priority than the upper ones (you need to select a rule in order to enable this button).

❖ **Control Style:**



The Control Style Manager designed to help you managing and designing the selected control, and it contains the following functions:

- **Control Style Properties:**
  - **W x H (px):** Set the width and the height in pixels for the selected control.
  - **Left x Top (px):** Set the left and top positions in pixels for the selected control.
  - **Background Color:** Set the background Color for the selected control.
  - **Text Color:** change the font color for the selected control.
  - **Text Size:** change the font size for the selected control.
  - **Text Align:** Align the text for the selected control to left/ center/ right.
  - **Bold:** Set the text to bold for the selected control.
  - **Italic:** Set the text to Italics for the selected control.
  - **Underline:** Set the text to Underline for the selected control.
  - **Vertical Align:** Align the vertical position of the Text property in the Label control to top/ middle/ bottom. This feature only visible for Label control.
- **Border Design:** To design the border of the selected control. Can edit the following properties:
  - **Same for all sides:** When you check this option, you will control all the four sides of the control borders with one change on the top border properties.
  - **Style:** Set border style for the selected control as solid/ dashed/ dotted/ ...
  - **Width:** Set the border width for the selected control.
  - **Color:** Set the border color for the selected control.

  Note: You can click on this icon  to remove the color properties for the control (it will become transparent).

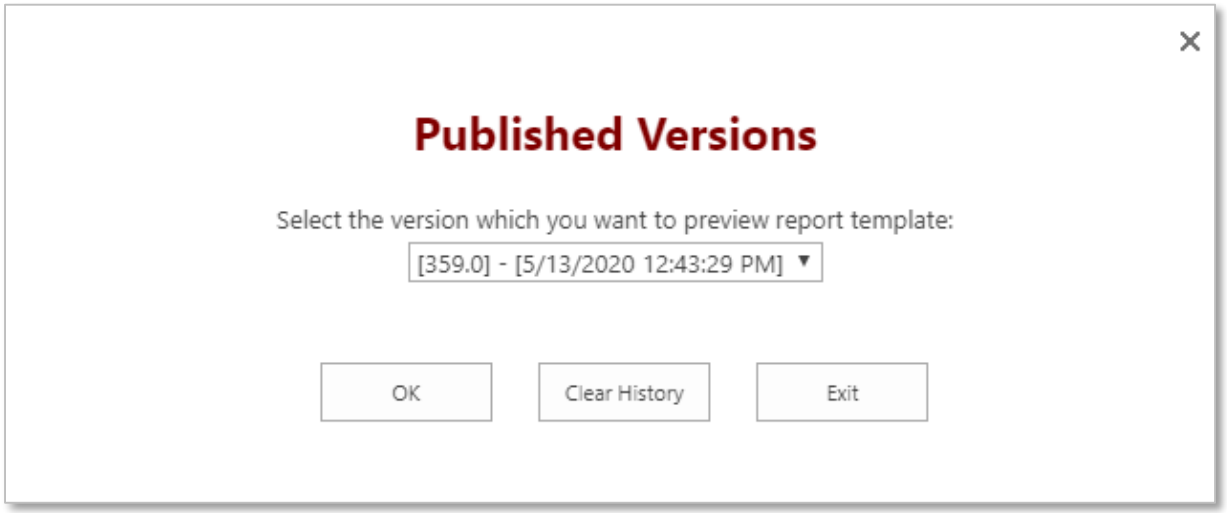
- **Delete:** delete the selected control(s).
- **Bring to front:** Brings the selected control to the front of the other controls.
- **Send to back:** Sends the selected control back behind the other controls.
- **Copy Format**: Copy the format of the current control. When clicking this button, the mouse cursor will change, this means you need to click on the target control to apply the copied format on it.
- **Clear Format**: Reset formatting of the selected control and return its style to the original state.

- **Design Group:**
  - **Run Report:** Clicking on **Run Report button** will display how the report will be rendered for users and will allow you to test it, check and validate its data and visual parts such as report table, columns, charts, format, style …etc.
  - **Query Builder**: Clicking on this button will open the **Query Builder** Manager section/dialog, for more information on the **Query Builder**, Please refer to the **Query Builder section** in this manual.
  - **Language:** Click on the Language button to change the Language of the report or copy the default language to a new form template design.

    SPARK Reports Builder provides the designer with the ability to design multilingual reports, these reports will be displayed for users depending on their operating systems or browsers preferred languages. **SPARK Report** supports all SharePoint supported languages with RTL-LTR directions. Just design the report using the primary language, change the report to the desired new language and translate the reports labels, table header's captions and charts labels accordingly.

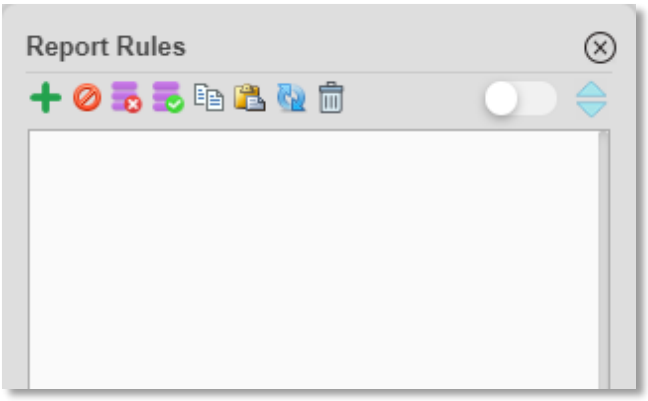
Auto-Populate: Clicking this button will do the following: The columns of the correspondent report's query will be automatically populated and built instantly in the report design workspace as a **Report Table** with all their properties and configuration settings. All populated columns will have headers captions, which represent their titles. The designer can continue designing and building the report as desired the way he/she wants after that.

- **Publishing Group:**
  - **Publish**: Please refer to Publishing a Report section.
  - **Unpublish**: Please refer to Unpublishing a Report Section.
  - **Versions:** Displays a list of all saved and published report's versions. From this list, a report can be rolled back to a specific earlier saved or published version. The user can click on **Clear History** button to clear all reports's published versions of the system except for the last ten; this will reduce the unnecessary storage size occupied by these versions. If the report has a large number of versions, it might take several minutes before completing the clearing process.



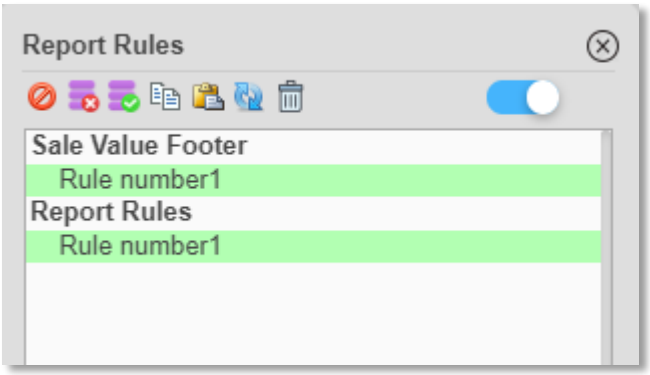
- **Settings Group:**
  - **Modify Source**: Extracts the source code of the report from the designer and updates the source of the report. This functionality is usually used for manual and external code editing and scripting.
  - **Report Rules/Form Rules:** When the user open the report design workspace the button caption will be **Report Rules**, while when the user open the Ad-Hoc Form design workspace the button caption will be **Form Rules**.

    To open the report rules panel, select the report you need to add rules to or the Ad-Hoc Form tab, you will see that the rules panel appears at the left side of the report design workspace.



By default, the rules panel will show only rules that are associated to the selected report or Form. The rules panel has the following buttons in order to work with rules associated with the selected report or form, these buttons are:

- **Show all:** By switching ON this toggle switch, you can view all related rules for the report/form and its controls. Double click on the rule will open the Rule Manager dialog to manage this rule the way you want.

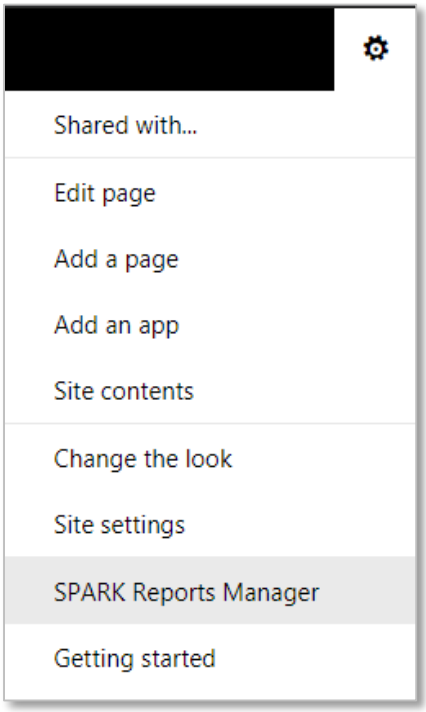


- **Create:** Create a new rule for the selected report/form.
- **Disable/Enable:** Disable or enable the selected rule on the selected report/form. If the selected rule is disabled, the enable icon will be shown, If the selected rule is enabled, the disable icon will be shown. Note that disabled rules will not be executed at runtime.
- **Enable All:** Enable all rules on the selected report/form.
- **Disable All:** Disabled all rules on the selected report/form.
- **Copy Rule:** Copy the selected rule for the report/form (you need first to select a rule in order to enable this button).
- **Paste Rule:** Paste the copied rule from another control/report/form to the current/selected report/form.
- **Refresh:** Refresh rules list in the selected report/form.
- **Delete Rule:** Delete the selected rule (you need to select a rule in order to enable this button).
- **Up:** Move the selected rule one level up in the rules list; uppers rules have higher priority than the down ones (you need to select a rule in order to enable this button).
- **Down:** Move the selected rule one level down in the rules list; lower rules have lower priority than the upper ones (you need to select a rule in order to enable this button).

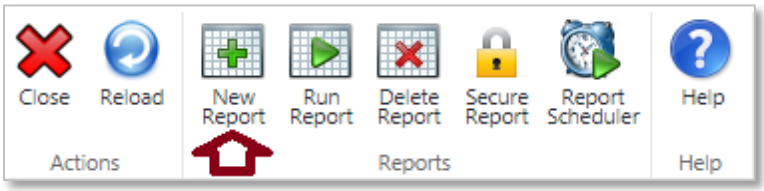
2 Creating a Report

2.1 Create a new Report

1. Click on **SPARK Reports Manager** on the site's top menu to open your site's reports management page. Note that this link will not be visible to you if you do not have at least "Manage lists" permission.



2. In the **Reports Manager** page, click on **New Report** button at the top ribbon of the page.



3. Start building your report by designing your query first using the **Query Builder**, then design your report based on the selected columns and query output. Refer to **Query Builder** section for more details.

Note: The designer's user account must have at least a "Manage lists" permission to be able to create and manage reports. Refer to **SPARK Reports Manager** for more details.

3 Shortcut keys

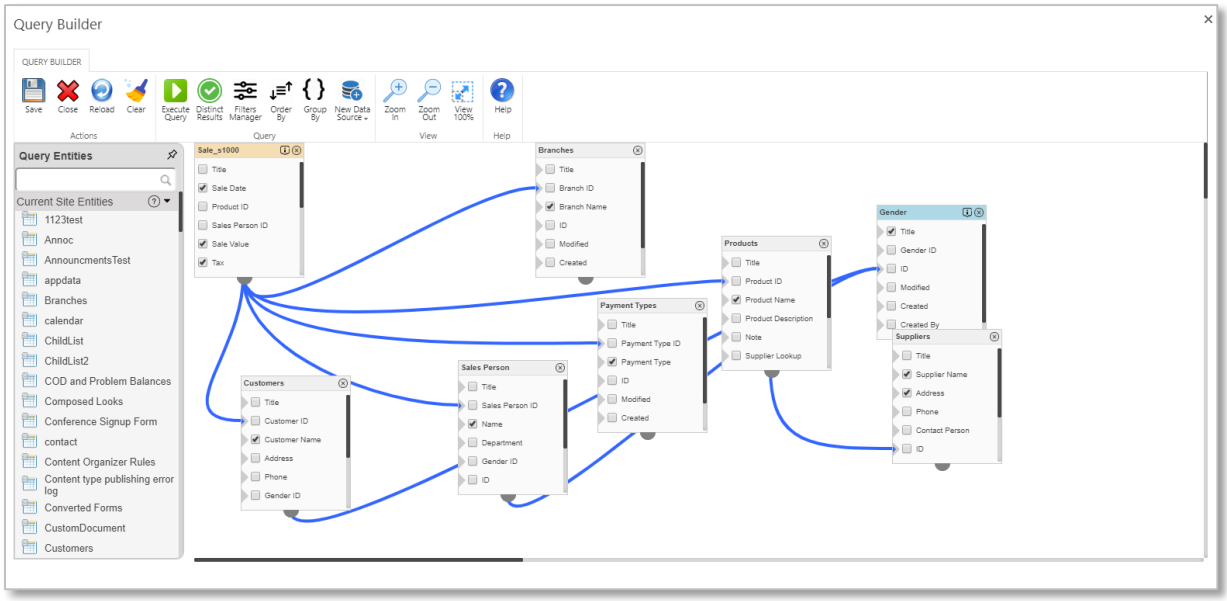
A shortcut key is a combination of keys that executes a specific function or command within the **SPARK Reports Builder**.
The followings are shortcut keys combinations that are available to be used when designing a report using **SPARK Reports Builder**:

Shortcut keys	**Description**
Ctrl + Z	Undo current form changes and return back one step
Ctrl + Y	Redo current form changes and restore them one step
Ctrl + A	Selects all controls on the current report page. Note: To make this shortcut working properly, you must select at least one control. Alternatively, you can click on "Select All" in the Edit Group at the design workspace top ribbon.
Ctrl + X	Cut the selected control(s) and put it into the clipboard
Ctrl + C	Copy the selected control(s) and put it into the clipboard
Delete	Delete the selected control(s)
Ctrl + S	Save the form
Ctrl + P	Publish the form

4 Query Builder

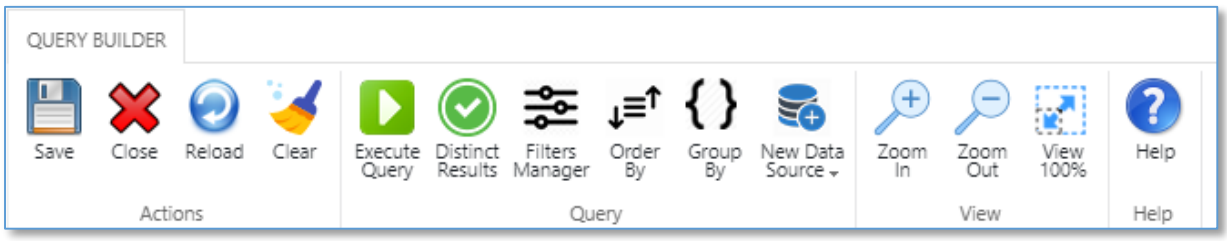
Note: Using data sources from other lists in the same web application or other web applications to build your report query applies to SPARK Reports Builder Standard and Enterprise Editions only. Using data sources from external databases such as (SQL/Oracle) to build report query applies to SPARK Reports Builder Enterprise Edition only.

In order to build your report using **SPARK Reports Builder** you need first to create, design and build your report query and specify columns you want to have in your report design.



4.1 Query Builder Ribbon

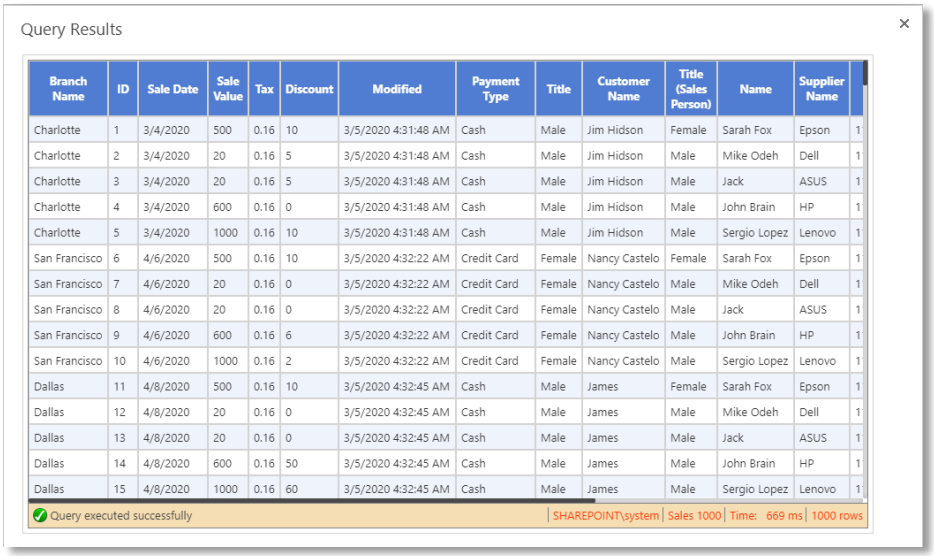
Located at the top of the **Query Builder** dialog page. Top ribbon groups and related buttons descriptions are listed below.



- **Actions Group:**
  - **Save:** Will save the current query design, data sources, filters, sorting, grouping by and its settings.
  - **Close**: Clicking **Close** will close the **Query Builder** dialog and return to the report design workspace's page. If there been changes on the report query design that have not been saved yet, a warning dialog will appear to confirm saving before closing the dialog. If the user confirm closing without saving the dialog, the unsaved query design will be lost.
  - **Reload**: Clicking **Reload** will reload/refresh the current query design to the last saved one.

- **Clear:** Clicking **Clear** button will reset and remove all query's design, configurations. Note that this will not affect the report design and its settings.

- **Query Group:**
  - **Execute Query**: Clicking this button will allow report designer to test his/her query on spot, all selected columns will be shown in a table dialog and will show all returned data. The table will have pages if the number of rows exceeded 300 rows, the designer can navigate between these pages by clicking on the page number at the bottom of the table. In addition, the designer can sort table's data "ascend/descend" by clicking on the column header. The Query Results dialog will show important information about the query execution at the bottom section: **Query Status, Executed By, Report Name, Query Execution Duration Time, and Number of returned rows.**

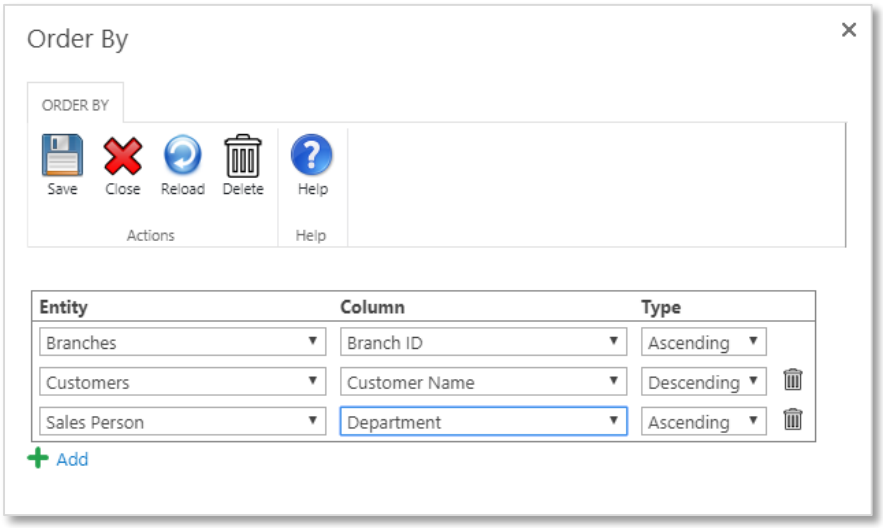

Query Results

Branch Name	ID	Sale Date	Sale Value	Tax	Discount	Modified	Payment Type	Title	Customer Name	Title (Sales Person)	Name	Supplier Name	
Charlotte	1	3/4/2020	500	0.16	10	3/5/2020 4:31:48 AM	Cash	Male	Jim Hidson	Female	Sarah Fox	Epson	1
Charlotte	2	3/4/2020	20	0.16	5	3/5/2020 4:31:48 AM	Cash	Male	Jim Hidson	Male	Mike Odeh	Dell	1
Charlotte	3	3/4/2020	20	0.16	5	3/5/2020 4:31:48 AM	Cash	Male	Jim Hidson	Male	Jack	ASUS	1
Charlotte	4	3/4/2020	600	0.16	0	3/5/2020 4:31:48 AM	Cash	Male	Jim Hidson	Male	John Brain	HP	1
Charlotte	5	3/4/2020	1000	0.16	10	3/5/2020 4:31:48 AM	Cash	Male	Jim Hidson	Male	Sergio Lopez	Lenovo	1
San Francisco	6	4/6/2020	500	0.16	10	3/5/2020 4:32:22 AM	Credit Card	Female	Nancy Castelo	Female	Sarah Fox	Epson	1
San Francisco	7	4/6/2020	20	0.16	0	3/5/2020 4:32:22 AM	Credit Card	Female	Nancy Castelo	Male	Mike Odeh	Dell	1
San Francisco	8	4/6/2020	20	0.16	0	3/5/2020 4:32:22 AM	Credit Card	Female	Nancy Castelo	Male	Jack	ASUS	1
San Francisco	9	4/6/2020	600	0.16	6	3/5/2020 4:32:22 AM	Credit Card	Female	Nancy Castelo	Male	John Brain	HP	1
San Francisco	10	4/6/2020	1000	0.16	2	3/5/2020 4:32:22 AM	Credit Card	Female	Nancy Castelo	Male	Sergio Lopez	Lenovo	1
Dallas	11	4/8/2020	500	0.16	10	3/5/2020 4:32:45 AM	Cash	Male	James	Female	Sarah Fox	Epson	1
Dallas	12	4/8/2020	20	0.16	0	3/5/2020 4:32:45 AM	Cash	Male	James	Male	Mike Odeh	Dell	1
Dallas	13	4/8/2020	20	0.16	0	3/5/2020 4:32:45 AM	Cash	Male	James	Male	Jack	ASUS	1
Dallas	14	4/8/2020	600	0.16	50	3/5/2020 4:32:45 AM	Cash	Male	James	Male	John Brain	HP	1
Dallas	15	4/8/2020	1000	0.16	60	3/5/2020 4:32:45 AM	Cash	Male	James	Male	Sergio Lopez	Lenovo	1

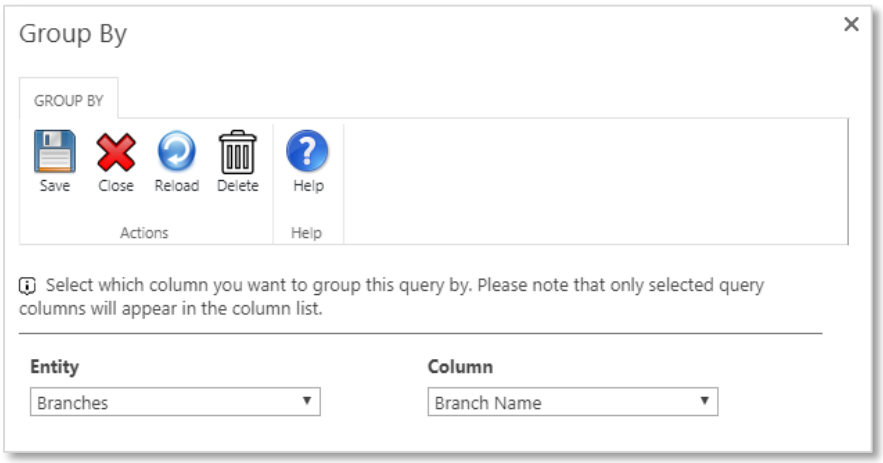
Query executed successfully | SHAREPOINT\system | Sales 1000 | Time: 669 ms | 1000 rows

  - **Distinct Results**: Checking this option will eliminate duplicate rows in returned query results data.
  - **Filters Manager**: Clicking this button will open the **Filters Manager** dialog in order to allow managing query filters by adding, editing and deleting them. Refer to the Filters Manager section for more details.
  - **Order By**: Clicking this button will open the "**Order By"** dialog in order to manage how the query results will be sorted. The designer can sort the query based on its columns, and can choose which columns will be sorted by and how they will be sorted (Ascending/Descending).
    In order to sort the query you need to select the **Entity** (list/Library/DB table) from the **Entity** Dropdown list, then select a column and the sorting type (Ascending/Descending), then click on **Save** button to save the query Order By criteria.
    Note: You can click on the **Add** icon 

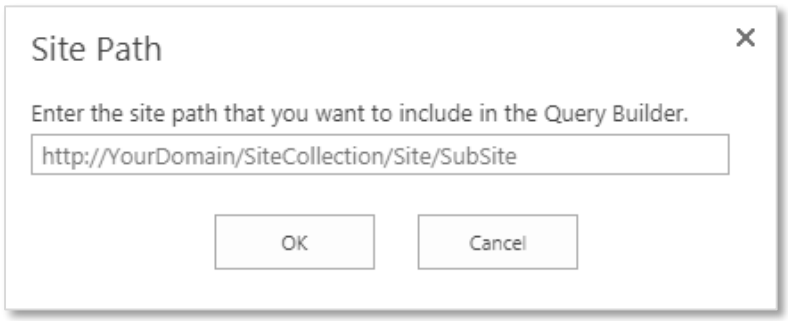
to add another sorting criterion.



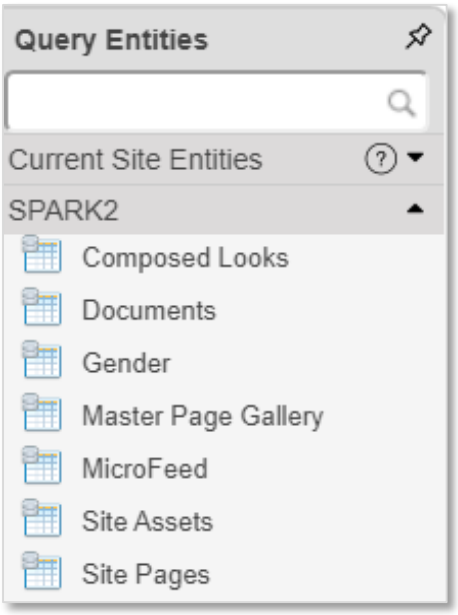
- **Group By**: Clicking this button will open the "**Group By"** dialog in order to manage query data aggregation. The designer can select which column he/she wants to group this query by. Please note that you can select multiple columns to group the query based on them.



- **New Data Source**: Clicking this button will open a submenu having the following two buttons:
  - **From SharePoint**: Clicking this button will open a dialog to allow the designer entering the site URL path that he/she wants to include in the **Query Builder** as a data source and includes its entities in the report query structure.

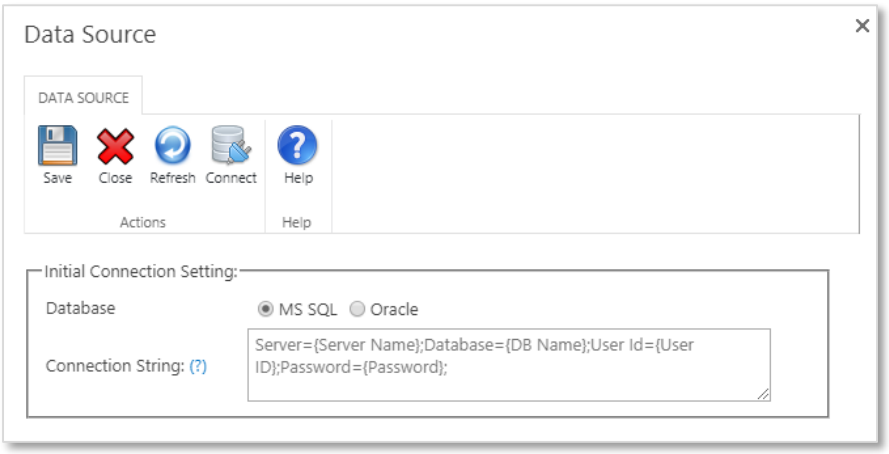


Once the designer clicks **OK** button, the Query builder will include the specified site and its entities in the **Query Toolbox**.

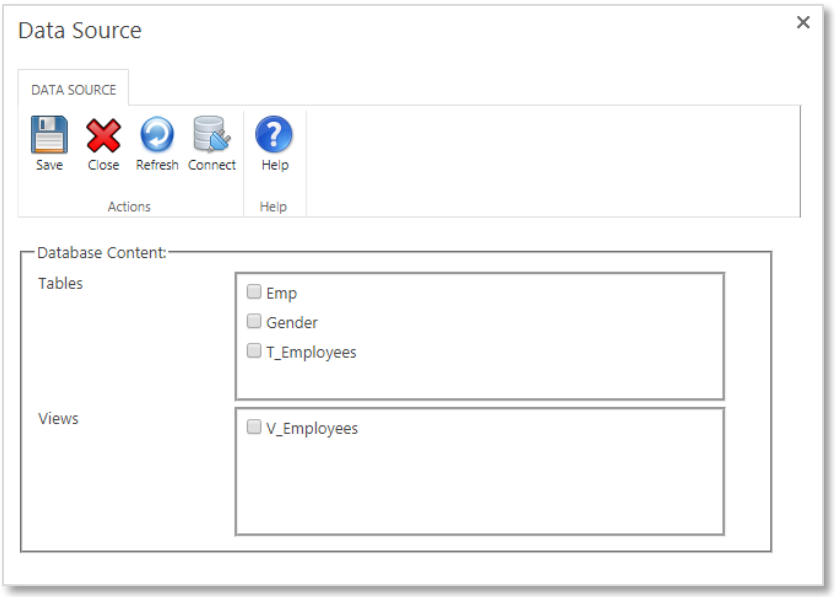


Note: The designer user account must have sufficient permissions to access the new data source's site, at least (read permission must be granted).

- **From Database**: Clicking this button will open a dialog to allow the designer specifying the external database (SQL/Oracle) that he/she wants to include in the **Query Builder** as a data source and includes its entities in the report query structure.



In order to add an external database as a data source to your query builder design, you need first to specify the type of the database (SQL/Oracle), Specify the **Connection String** to connect to the database. For more information about the connection string types and properties, refer to the following link: Connection Strings.
Click on the **Connect** button at the top ribbon of the dialog to connect to the database and retrieves its entities structures. The connected database entities will be distributed into two categories (**Tables and Views**), you can select which entities to include in your query by checking them in the dialog and clicking the **Save** button.

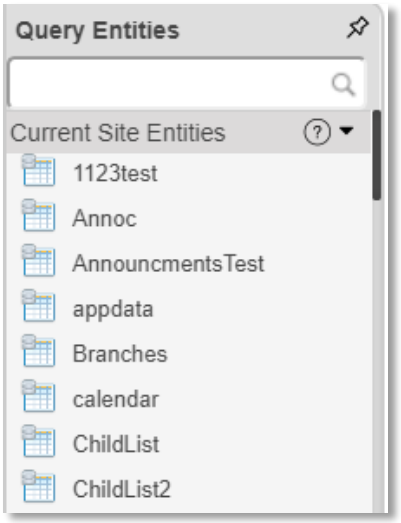


- **View Group:**
  - **Zoom In:** Clicking **Zoom In** will increase the size of the query diagram (query design workspace canvas).
  - **Zoom Out:** Clicking **Zoom Out** will decrease the size of the query diagram (query design workspace canvas).
- **View 100%:** Clicking **View 100%** will return the size of the query diagram (query design workspace canvas) to the original viewing size.

4.2 Create a Report Query

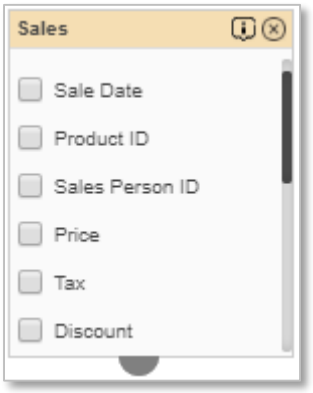
For General and History Audit Reports types only

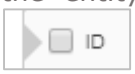
1. Click on the **Query Builder** button at the top ribbon of the report's design workspace page to open the Query Builder dialog.
2. By default, the **Query Builder** retrieves all current site's entities (lists and libraries) in the **Query Entities** panel; you can search for any entity using the search box in the panel.



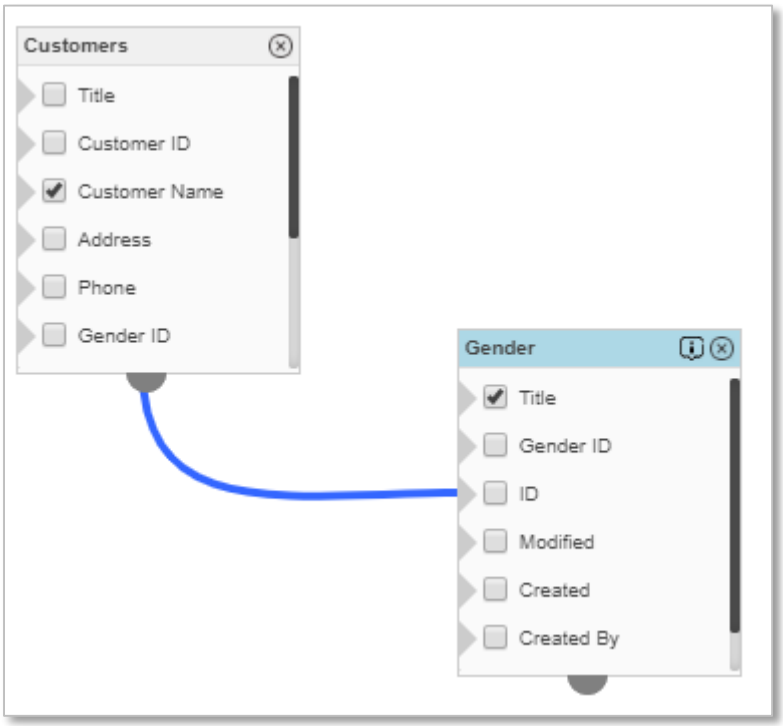
3. Drag the entity you want from the **Query Entities** panel and drop it in the query design workspace. You will have the entity structure box, which has the entity's

columns. You can drag/drop multiple entities in the design workspace in order to create relations between them.

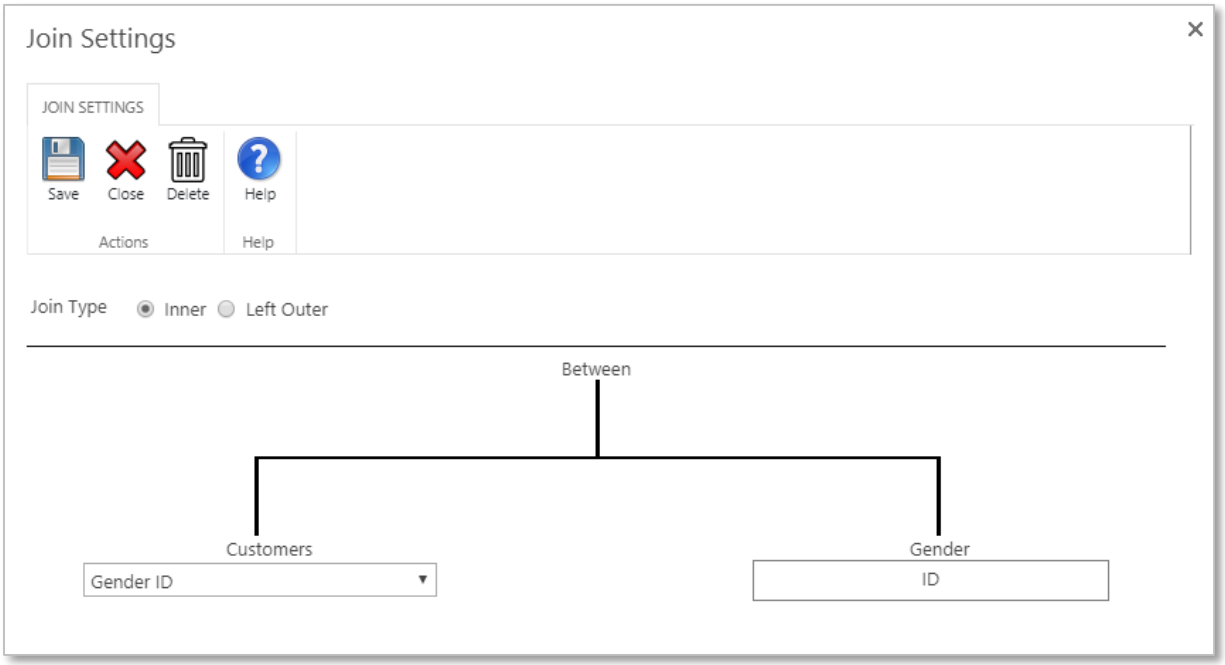


4. Select columns that you want to have in your report and create relations between entities by dragging the half rounded relation element located at the bottom of the entity structure box  and drop it on the other entity column 

i.e. Not applicable for **History Audit report** type.



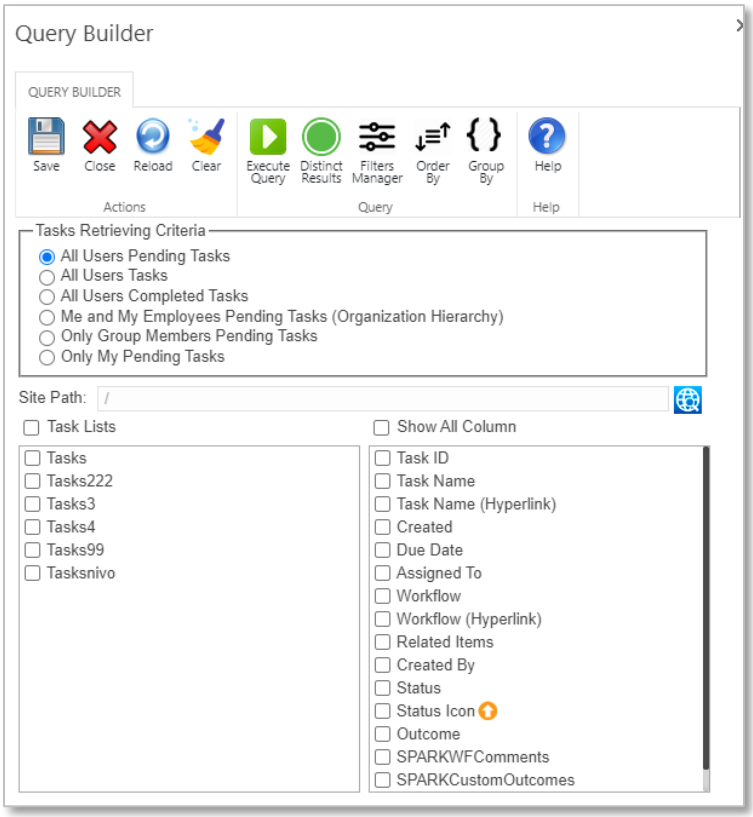
5. You can specify the relation type and columns when creating a relation between two entities. There are two available "**Join Types**", which you can select from: "**Inner and Left Outer**" join types.
i.e. Not applicable for **History Audit report** type.



6. You can execute the query and test returned data and duration time.
7. Save the query design and return to the report design workspace to design your report based on this query.

For Workflow Tasks and Key Performance Indicator Reports types only

1. Click on the **Query Builder** button at the top ribbon of the report's design workspace page to open the Query Builder dialog.
2. By default, the **Query Builder** will show the **Tasks Retrieving Criteria** to choose from.



3. You can choose the following types of workflow-tasks query criteria: All Users Pending Tasks, All Users Tasks, All Users Complete Tasks, Me and My Employees Pending Tasks (Organization Hierarchy), Only a specific Group Members Tasks and Only My "Current Logged in User" Pending Tasks.
4. Select the workflow tasks lists you want to include in the report from the Tasks Lists section in the dialog. You need to select at least one list to be able to save the query.
5. Select columns you want to have in the report from the columns list or you can select **Show All Columns** to show all the lists columns in the report's table.
6. Save the query design and return to the report design workspace to design your report based on this query.

See also

Reference

Other resources

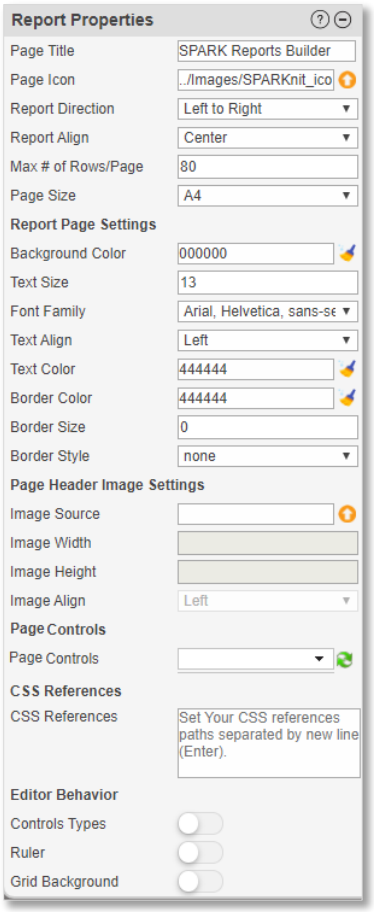
Video: https://youtu.be/it0MLx-Zng8 This video will show how to create a report's query, design and publish it in 5 minutes.

5 Report Workspace Properties

5.1 Report Properties

General Properties:

- **Page Title:** Specify the report's browser tab caption. The default value will be "SPARK Reports Builder".
- **Page Icon:** Specify the icon image that will appear in the browser tab. The default icon image will be SPARK Reports Builder icon .
- **Report Direction:** Specify the text direction of all controls in the report (**Left to Right or Right to Left**).
- **Report Align:** Specify the report alignment in the page at runtime (**Left, Right or Center**).
- **Max # of Rows/Page:** Specify the maximum number of rows that will be display per page in the **Report Table** controls on the report. This property will affect all **Report Table** controls in the report.
- **Page Size:** Specify the report's page size in which the report layout will be limited to. You can select from the following page sizes (**A0, A1, A2, A3, A4, A5, B0, B1, B2, B3, B4, B5, B6, C0, C1, C2, C3, C4, C5, Letter and Custom** "you need to specify the height and width of the page when selecting this option").
  Note that in case of selecting **Custom** page size; the report will not be suitable for printing out or exporting in case its generated data exceeds a single page count. This option is mostly used when creating dashboards, visual indicators or charts to be embedded into SharePoint pages using **SPARK Reports Viewer web part**.



Report Page Settings Properties:

- **Background Color:** Specify the background color of the **Report Page**. Note that you can click on this icon to remove the color property (it will become transparent).
- **Text Size:** Specify the font size of the **Report Page**.
- **Font Family:** Specify the font family of the **Report Page**.
- **Text Align:** Align the text of the **Report Page** to (Left/ Center/ Right).
- **Text Color:** Specify the font color of the **Report Page**. Note that you can click on this icon to remove the color property (it will become transparent).
- **Border Color:** Set the border color of the **Report Page**. Note that you can click on this icon to remove the color property (it will become transparent).
- **Border Size:** Set the border size of the **Report Page**.
- **Border Style:** Apply border style of the **Report Page** as solid/ dashed/ dotted/ ...etc.

Page Header Image Settings Properties:

- **Image Source:** Specify header image's URL or upload it using the upload icon .
- **Header Width:** Specify the width of the header image of the **Report Page**.
- **Header Height:** Specify the height of the header image of the **Report Page**.

- **Header Align:** Align the header image to (left/ center/ right) in the **Report Page**.

Report Page Controls:

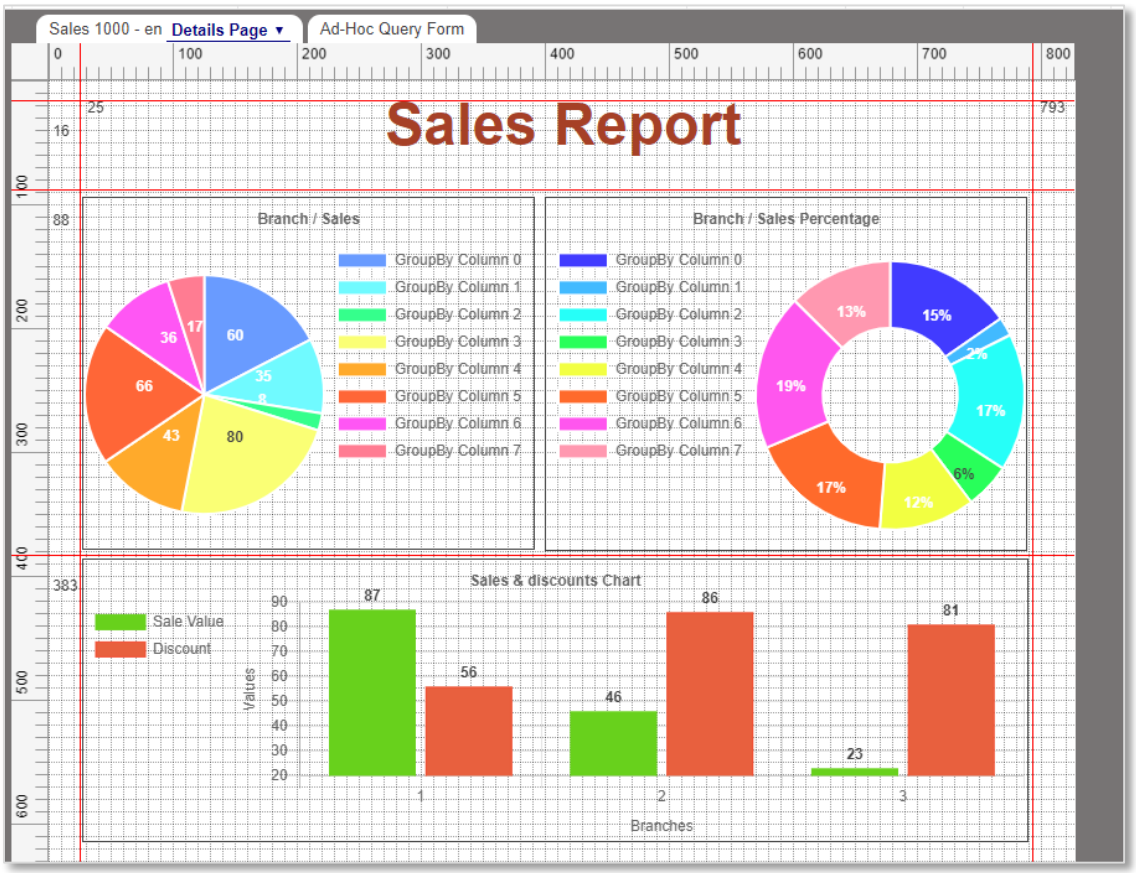
- **Page Controls:** Use this property to show Control's Properties for the selected control. You can click on the Refresh Button near the **Page Controls** to refresh the related controls of the **Report Page**. You can search for a control by typing its name or part of its name.

CSS References:

- **CSS References:** Use this property to set CSS files reference's paths in order to use their "CSS classes" on the report's page or controls styles.

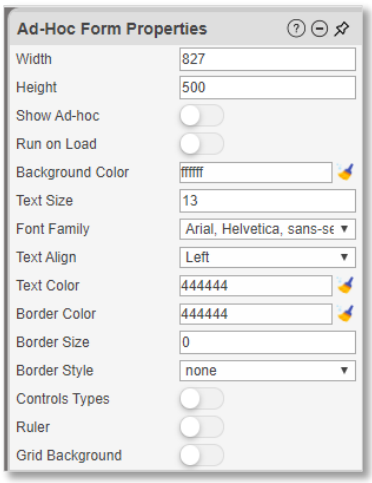
Editor Behavior:

- **Controls Types:** This property is for showing or hiding tooltips on the controls in the report's design workspace, these tooltips identify controls types when the mouse cursor moves over these controls, this will help the designer to figure out what types of controls he/she is dealing with during the designing phase.
- **Ruler:** Use this property to hide or show the ruler in the report's design workspace. The main advantage of the ruler is to help the designer in aligning controls both ways horizontally and vertically. When the selecting the control, the corresponding area of the ruler will be shaded automatically.

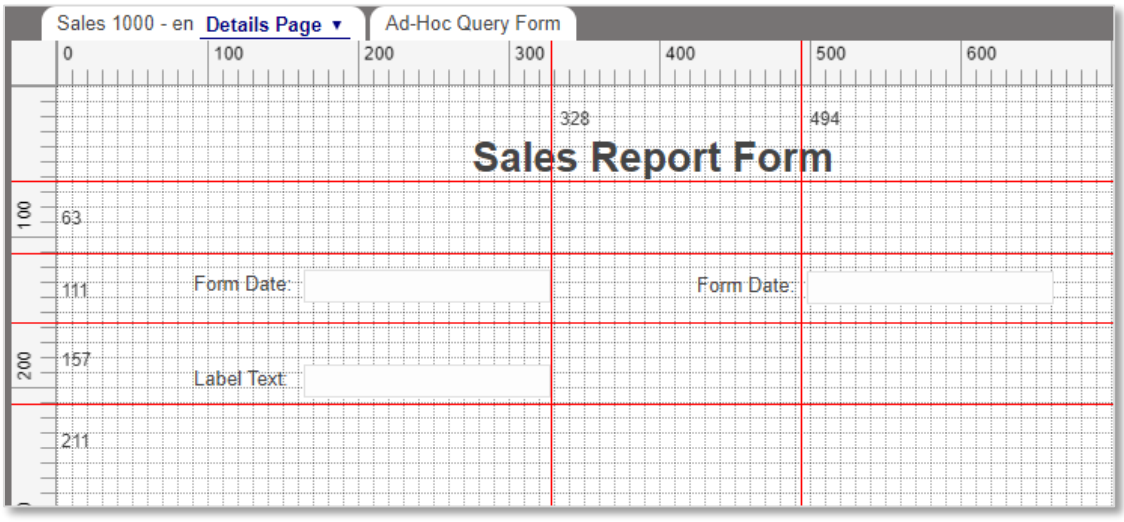


- **Grid Background:** This property is for showing or hiding the grid background of the **Report Page** in the report's design workspace.

5.2 Ad-Hoc Query Form Properties



- **Width:** Specify the Ad-Hoc Query form's width in pixels.
- **Height:** Specify the Ad-Hoc Query form's height in pixels.
- **Show Ad-Hoc:** Specify if the **Ad-Hoc Query Form** will be displayed in the form page when the report loads or when executing the query form. The default behavior is "hidden".
- **Run on Load:** Specify if the Ad-Hoc form will execute when running/loading the report. This property is switched "OFF" by default.
- **Background Color:** Specify the background color of the **Ad-Hoc Query Form**. Note that you can click on this icon to remove the color property (it will become transparent).
- **Text Size:** Specify the font size of the **Ad-Hoc Query Form**.
- **Font Family:** Specify the font family of the **Ad-Hoc Query Form**.
- **Text Align:** Align the text of the **Ad-Hoc Query Form** to (Left/ Center/ Right).
- **Text Color:** Specify the font color of the **Ad-Hoc Query Form**. Note that you can click on this icon to remove the color property (it will become transparent).
- **Border Color:** Set the border color of the **Ad-Hoc Query Form**. Note that you can click on this icon to remove the color property (it will become transparent).
- **Border Size:** Set the border size of the **Ad-Hoc Query Form**.
- **Border Style:** Apply border style of the **Ad-Hoc Query Form** as solid/ dashed/ dotted/ ...etc.
- **Controls Types:** This property is for showing or hiding tooltips on the controls in the form's design workspace, these tooltips identify controls types when the mouse cursor moves over these controls, this will help the designer to figure out what types of controls he/she is dealing with during the designing phase.
- **Ruler:** Use this property to hide or show the ruler in the form's design workspace. The main advantage of the ruler is to help the designer in aligning controls both ways horizontally and vertically. When the selecting the control, the corresponding area of the ruler will be shaded automatically.



- **Grid Background:** This property is for showing or hiding the grid background of the **Ad-Hoc Query Form** in the report's design workspace.

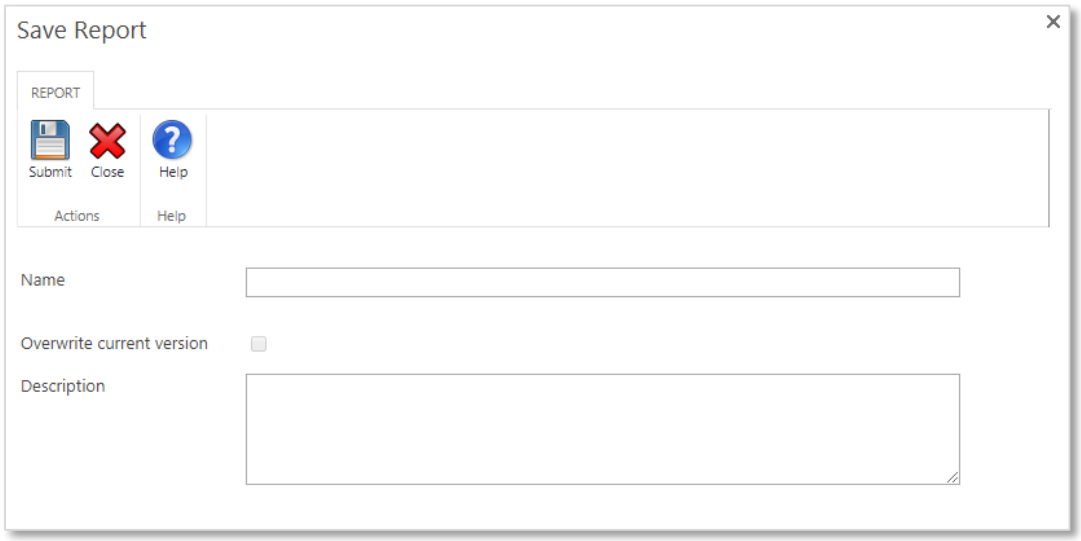
6 Saving and Publishing Reports

A report must be published in order to make it available to end users. Published reports are automatically saved. You can use the **Save** button to save changes to the report design and query without publishing it.

6.1 Saving a Report

To save a report:

1. In SPARK Report's design workspace page top ribbon, click the **Save** button.
2. Clicking the **Save** button will open the following popup dialog:



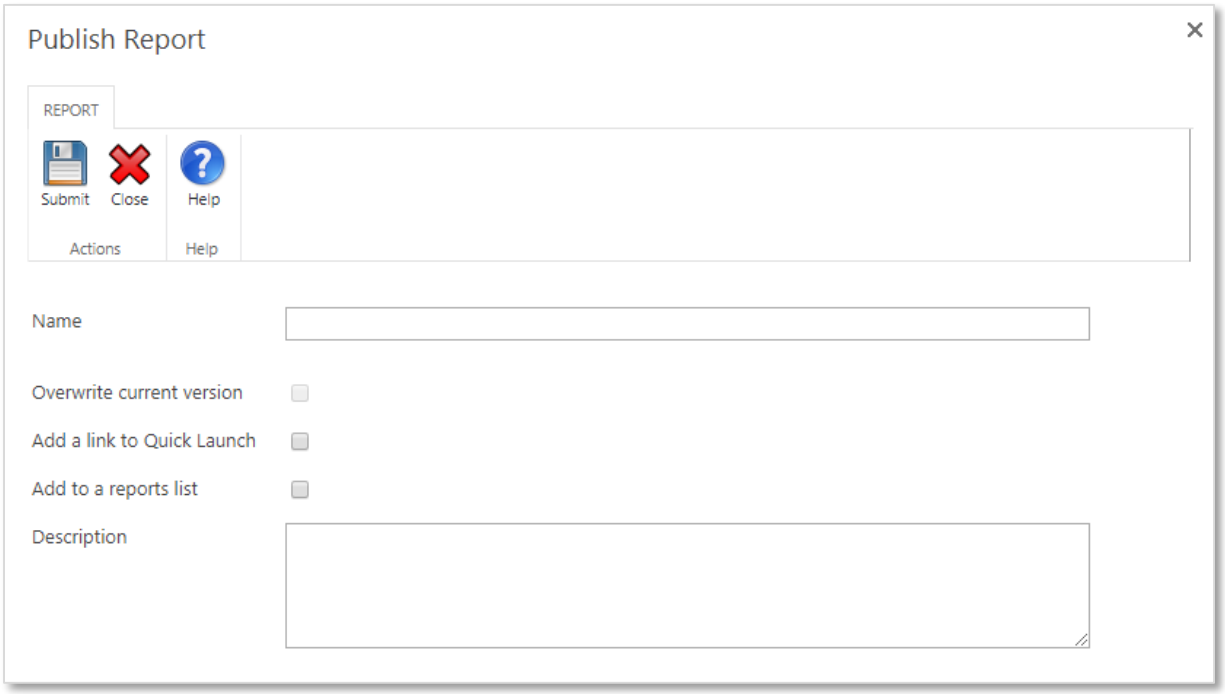
   - **Submit:** Click on **Submit** button will save the current report design and generate an unpublished version of it.
   - **Name:** Specify the name of the report. This is a mandatory and a unique property, which means it must be entered for any newly created report and the name must not be assigned for any other report across the site.
   - **Overwrite current version:** You can optionally choose to "Overwrite" any existing version of any currently existing report.
   - **Description:** Optionally specify a brief description of the report.
3. Click **Submit** button at the top ribbon of the **Save Report** dialog.
4. A progress dialog will display while saving the report.
5. A confirmation message will appear to indicate that the saving process has been completed successfully. Click **OK** to close the dialog.
6. If there is any problem in saving the report, an error dialog will appear, showing the error type and message. Try saving the report again, if the error keeps occurring then contact your system administrator or our technical support.

6.2 Publishing a Report

Before a report becomes available to end users on the SharePoint site, it must be published.

To publish a report:

1. Click the **Publish** button in **SPARK Reports Builder** top ribbon.
2. Clicking the **Publish** button will open the following popup dialog:



- **Submit:** Click on **Submit** button will save the current report design and generate an published version of it.
- **Name:** Specify the name of the report. This is a mandatory and a unique property, which means it must be entered for any newly created report and the name must not be assigned for any other report across the site.
- **Overwrite current version:** You can optionally choose to "Overwrite" any existing version of any currently existing report.
- **Add a link to Quick Launch:** Enabling this option will add a link of the published report to the **Quick Launch** menu of the current site.
- **Add to a reports list:** Enabling this option will create a report links list in the site and add this report's link to it along with the report description.
- **Description:** Optionally specify a brief description of the report.

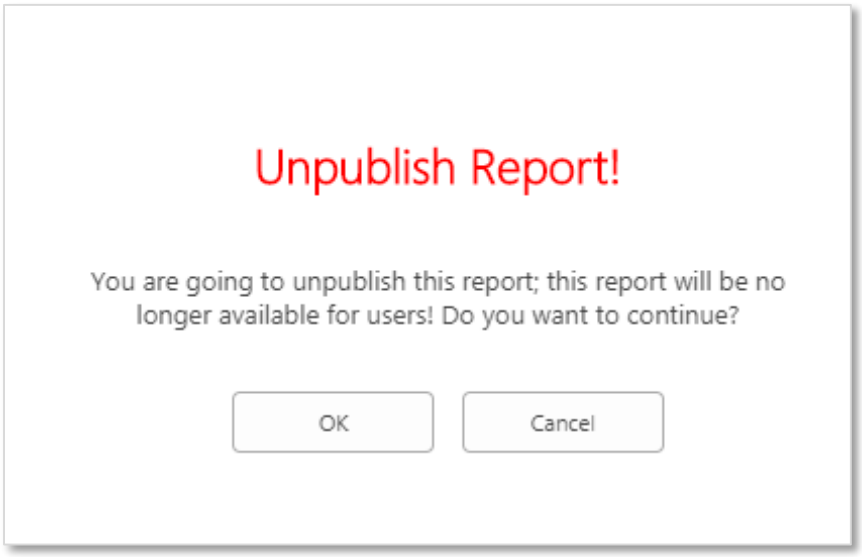
3. Click **Submit** button at the top ribbon of the **Publish Report** dialog.
4. A progress dialog will display while publishing the report.
5. A confirmation message will appear to indicate that the publishing process has been completed successfully. Click **OK** to close the dialog.
6. If there is any problem in publishing the report, an error dialog will appear, showing the error type and message. Try publishing the report again, if the error keeps occurring then contact your system administrator or our technical support.

6.3 Unpublishing a Report

This will undo the publishing process on the report, remove report's links on the site and the report will no long be available for users to use.

To unpublish a report:

1. Click the **Unpublish** button in **SPARK Reports Builder** top ribbon.
2. Clicking the **Unpublish** button will open the following popup dialog:



3. Click **OK** button to unpublish the report.
4. A progress dialog will display while unpublishing the report.
5. A confirmation message will appear to indicate that the unpublishing process has been completed successfully. Click **OK** to close the dialog.
6. If there is any problem in unpublishing the report, an error dialog will appear, showing the error type and message. Try unpublishing the report again, if the error keeps occurring then contact your system administrator or our technical support.

7 Running a Report

To run a report you can click on the **Run Report** button at the top ribbon of the report design workspace's page. A new browser tab having the report's viewer page will open and the report will be rendered in that browser tab the same as it will be rendered to site's users when they run the report.

7.1 Report Viewer Ribbon's Buttons

SPARK Report Viewer Ribbon is located at the top side of the Viewer page. The following is a full description about each button in this ribbon.

- **First Page**: Open the first page of the report.
- **Previous Page**: Open the previous page of the report.
- **Next Page:** Open the next page of the report.
- **Last Page**: Open the last page of the report.
- **Zoom In**: Clicking **Zoom In** will increase the size of the report's view.
- **Zoom Out:** Clicking **Zoom Out** will decrease the size of the report's view.
- **Full View**: Clicking **Full View** will return the size of the report's view to the original/normal viewing size.
- **Save As**: Clicking this button will export the report along with its all pages to a file, you can choose what file format you want to export the report to: **PDF, Word, CSV, Excel with/out style**.
- **Print**: Clicking this button will print out the report with its all pages to a local or a network printer. A print preview dialog will show first before printing.
- **Run Report**: Clicking this button will validate and execute the **Ad-Hoc Query Form** and pass entered parameters to the report in order to generate a dynamic report query.
- **Reload**: Clicking this button will reload/refresh the report's page.
- **Back**: Clicking this button will redirect the report's page to the back/source page.

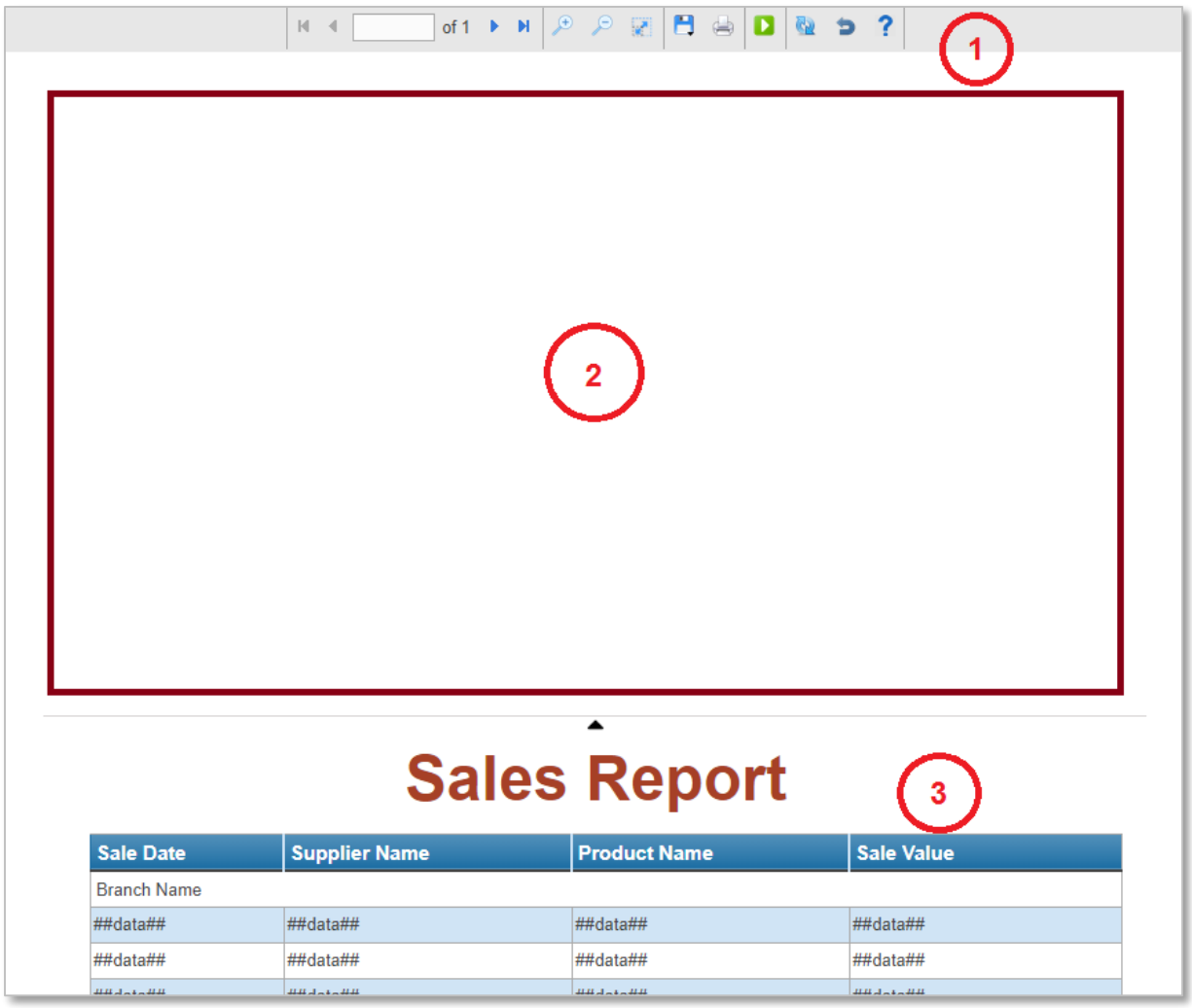
7.2 Report Viewer Structure

SPARK Report Viewer consists of three major parts:

1. **Report Viewer top ribbon:** This part contains buttons to work with the report.

2. **Ad-Hoc Query Form:** This part contains the Ad-Hoc Form in which the user runs a dynamic query and get report' data based on the Form's parameters. This form can be shown or hidden by clicking this icon .

3. **Report Details:** This part contains report's information and data such as report tables, charts, calculated parts, pages and footers.



8 Print Out a Report

SPARK Reports Builder allows the user to print out a report at runtime using the report viewer. When the user clicks on the Print button, it shows a progress dialog, which renders the reports pages in memory and sends them to the browser's print preview dialog. The user will be able to check out the report on the print preview dialog before printing it out to a local or network printer.

While **SPARK Reports Builder** works fine on all major browsers, we do recommend using **Chrome** browser for the best performance and report's features utilization.

9 Report Controls

SPARK Report Builder provides users with many various controls to design outstanding reports without limitations and to support creativity and scalability in representing SharePoint and external sources' data to users and decision makers.

9.1 General Controls

9.1.1 Report Table Control

The **Report Table** control represents the report's data in tabular way; it displays a grid of columns and rows of the report's query resulted data. The designer will be able to drag columns from the Toolbox and drop the on this control, sort them, add formatting and formula rules to columns/rows, create calculated columns, Create table footer, design its style …etc.

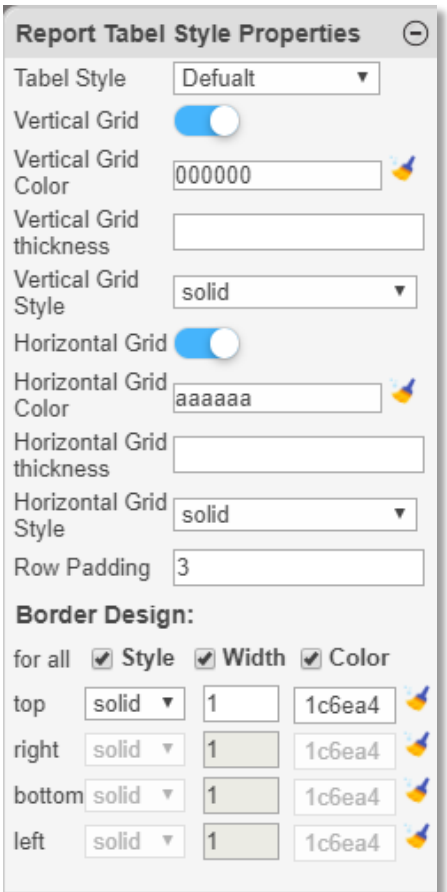
Control Properties:

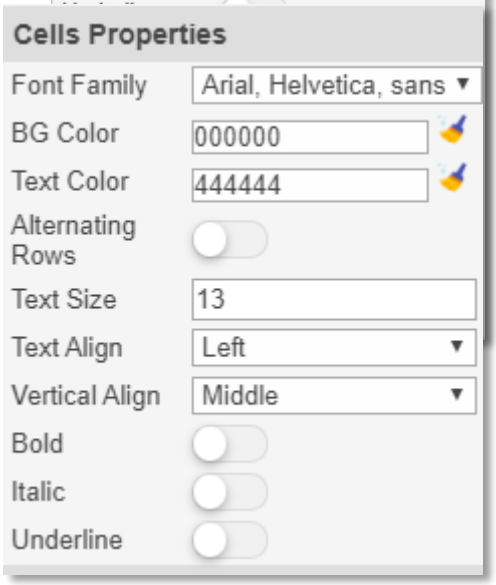
- **Control Type:** The type of the selected control.
- **ID:** The Identity of the control. The ID is a unique value, no duplicate IDs allowed in the report.
- **Table Title:** Set a **Report Table** title, which will be displayed on the top of the control.
- **Tooltip:** Appears on running the report and the user moves the mouse pointer over this control.
- **CSS Classes:** The HTML class attribute makes it possible to define equal styles for elements with the same class name.
- **Report Table Style:** Clicking on this button will show the **Report Table Style Properties** panel, which has many properties to adjust control's elements style, such as (Ready-made style to choose from, Header Properties, Cell Properties, Footer Style Properties, and Table Title Properties). Refer to **Report Table Style** section for more details.
- **Columns Manager:** Clicking on this button will show the **Columns Manager** panel, which allow the designer to manage **Report Table** columns, sort them out, configure their styles, rules, visibility and create calculated columns. Refer to **Columns Manager** section for more details.
- **Footer Cells Manager:** Clicking on this button will show the **Footer Cells Manager** panel, which allow the designer to manage **Report Table** footer cells, configure their styles, rules and create calculated footer cells. Refer to **Footer Cells Manager** section for more details.
- **Aggregated Footer:** This property will be visible if the report's query is a group by query "Aggregated"; in this case, the designer can switch this property **ON** to have the Footer section repeated in every grouped data and get calculations for each one separately.
- **Style:** The HTML style code in a free style editor that can be modified on the spot for the selected control.
- **Hidden:** Specifies if the control will be visible or not when running the report.

9.1.1.1 Report Table Style

The **Report Table Style Manager** designed to help you managing and designing the selected **Report Table** control. It contains the following properties:

- **General Properties:**
  - **Table Style:** Specify a table style by selecting form built-in table styles.
  - **Vertical Grid:** Show/hide vertical lines on the report table control.
  - **Vertical Grid Color:** Specify the vertical lines' color on the report table control.
  - **Vertical Grid Thickness:** Specify the vertical lines' thickness on the report table control. Accepts numeric value only.
  - **Vertical Grid Style:** Specify the vertical style on the report table control. [dashed, dotted, groove, hidden, inherit, inset, none, outset, ridge and solid]
  - **Horizontal Grid:** Show/hide horizontal lines on the report table control.
  - **Horizontal Grid Color:** Specify the horizontal lines' color on the report table control.
  - **Horizontal Grid Thickness:** Specify the horizontal lines' thickness on the report table control. Accepts numeric value only.
  - **Horizontal Grid Style:** Specify the horizontal style on the report table control. [dashed, dotted, groove, hidden, inherit, inset, none, outset, ridge and solid]
  - **Row Padding:** Set the row padding value. Accepts numeric value only.
  - ❖ **Border Design:** To design the border of the selected control. Have the following properties:
  - **for all:** When you check this option, you will control all the four sides of the control borders with one change on the top border properties.
  - **Style:** Set border style for the selected report table control as solid/ dashed/ dotted/ ...
  - **Width:** Set the border width for the selected report table control.
  - **Color:** Set the border color for the selected report table control.
- **Header Properties:**
  - **BG Color:** Set the background Color for the selected report table control's header.
  - **Font Family:** Specify the text font family [Arial, Serif…etc.] for the





selected report table control's header.
- **Text Color:** Specify the font color for the selected report table control's header.
- **Text Size:** Set the font size for the selected report table control's header. Accepts numeric value only.
- **Text Align:** Align the text for the selected report table control's header to left/ center/ right.
- **Vertical Align:** Align the vertical position of the Text property for the selected report table control's header to top/ middle/ bottom.
- **Bold:** Set the text to bold for the selected report table control's header.
- **Italic:** Set the text to Italics for the selected report table control's header.
- **Underline:** Set the text to Underline for the selected report table control's header.

❖ **Border Design:** To design the border of the selected control's header. Have the following properties:

- **for all:** When you check this option, you will control all the four sides of the control's header borders with one change on the top border properties.
- **Style:** Set border style for the selected report table control's header as solid/ dashed/ dotted/ ...
- **Width:** Set the border width for the selected report table control's header.
- **Color:** Set the border color for the selected report table control's header.

- **Cells Properties:**
  - **Font Family:** Specify the text font family [Arial, Serif…etc.] for the selected report table control's cells.
  - **BG Color:** Set the background Color for the selected report table control's cells.
  - **Text Color:** Specify the font color for the selected report table control's cells.
  - **Alternating Rows:** Enable/disable alternating rows.
  - **Alternating BG Color:** Specify the background Color for the selected report table control's alternating rows' cells.
  - **Alternating Text Color:** Specify the font color for the selected report table control's alternating rows' cells.
  - **Text Size:** Set the font size for the selected report table control's cells. Accepts numeric value only.
  - **Text Align:** Align the text for the selected report table control's cells to left/ center/ right.
  - **Vertical Align:** Align the text for the selected report table control's cells vertically to Middle/ Top/ Bottom.
  - **Bold:** Set the text to bold for the selected report table control's cells.
  - **Italic:** Set the text to Italics for the selected report table control's cells.
  - **Underline:** Set the text to Underline for the selected report table control's Cells.
- **Footer Style Properties:**
  - **Font Family:** Specify the text font family [Arial, Serif…etc.] for the selected report table control's footer.
  - **BG Color:** Set the background Color for the selected report table control's footer.

- **Text Color:** Specify the font color for the selected report table control's footer.
- **Text Size:** Set the font size for the selected report table control's footer. Accepts numeric value only.
- **Text Align:** Align the text for the selected report table control's footer to left/ center/ right.
- **Vertical Align:** Align the vertical position of the Text property for the selected report table control's footer to top/ middle/ bottom.
- **Bold:** Set the text to bold for the selected report table control's footer.
- **Italic:** Set the text to Italics for the selected report table control's footer.
- **Underline:** Set the text to Underline for the selected report table control's footer.

❖ **Border Design:** To design the border of the selected control's footer. Have the following properties:

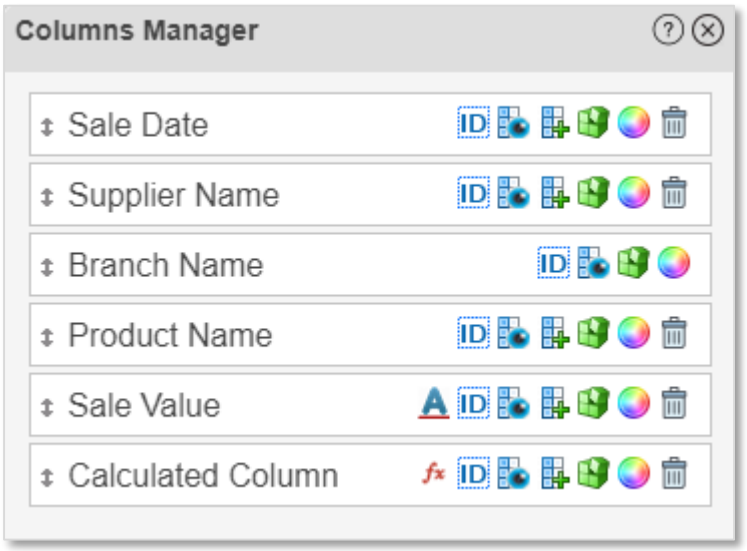
- **for all:** When you check this option, you will control all the four sides of the control's footer borders with one change on the top border properties.
- **Style:** Set border style for the selected report table control's footer as solid/ dashed/ dotted/ ...
- **Width:** Set the border width for the selected report table control's footer.
- **Color:** Set the border color for the selected report table control's footer.

- **Table Title Properties:**
  - **Text Color:** Specify the font color for the selected report table control's title.
  - **Text Size:** Set the font size for the selected report table control's title. Accepts numeric value only.
  - **Text Align:** Align the text for the selected report table control's title to left/ center/ right.
  - **Title Side:** Align the vertical position of the Text property for the selected report table control's title to top/bottom.
  - **Bold:** Set the text to bold for the selected report table control's title.
  - **Italic:** Set the text to Italics for the selected report table control's title.
  - **Underline:** Set the text to Underline for the selected report table control's title.

Note: You can click on this icon to remove the color properties for the control (it will become transparent).

9.1.1.2 Columns Manager

The **Report Columns Manager** designed to help you managing and designing the selected **Report Table** control's columns. It contains the following parts:



- **Column Section:** The container of each column in the Report Table control.

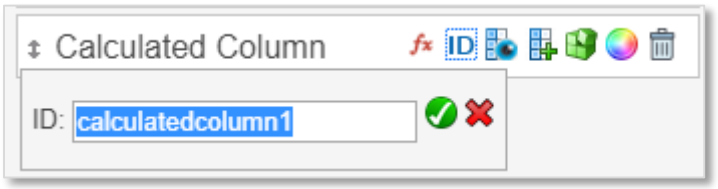
  

  You can move each **Column Section** by the mouse, up and down to change its position in the report design. In addition, you can click on the column caption and change it the way you want and it will be reflected directly on the selected Report Table columns header.

- **Delete Column Button:** Clicking this button will delete the column. A confirmation message will appear before proceeding with the process.
- **Column Style Button:** Clicking this button will open **the Column Style Manager** dialog. Refer to the **Change Column Style** section for more details.
- **Manage Column Rules Button:** Clicking this button will open **the Column Rules Manager** dialog. Refer to the **Rules** section for more details.
- **Insert Calculated Value Column Button:** Clicking this button will insert a new **Calculated Column** to the Report Table design next to the column you clicked this button on, also it will add a new Column Section in the **Columns Manager** beneath the column you clicked this button on. Refer to the **Calculated Column** section for more details about this type of columns.
- **Hide Column Button:** Clicking this button will hide the column in the selected Report Table control at runtime. The column will still be visible in the **Report Design Workspace** and the button will change to **Show Column Button** in order to allow showing it again if the designer decided to.

- **Edit Column ID:** Clicking this button  will allow editing the internal column ID, which is used in rules and formulas. Once you change the ID, click on the  button to apply the change.



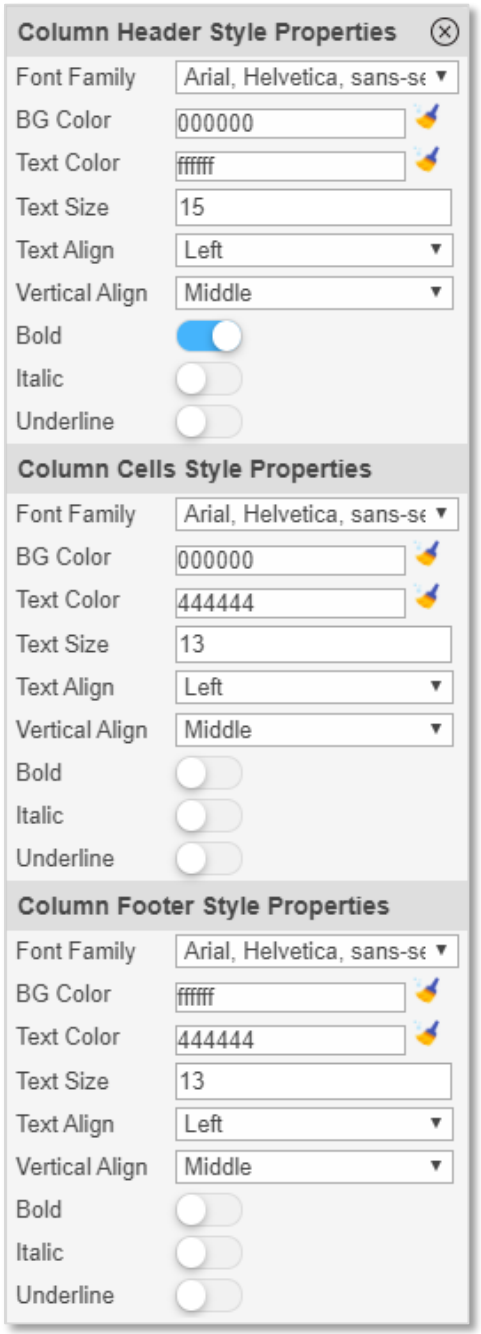
- **Calculated Column Formula (*fx*) Button:** (**For Calculated Columns only**). Clicking this button  will open **the Calculated Column Formula Manager** dialog. Refer to the **Calculated Column Formula** section for more details.
- **Column Format:** Clicking this button  will allow you to change the column's format that will be reflected on the selected **Report Table** control at runtime. This button will appear only for specific List/Library columns types such as:

SharePoint Column Type	Formats
Boolean	**Numeric | Text | Graphic**
Currency	**Numeric | Currency**
Number	**Numeric | Percent**
Hyperlink	**Text | Link**
Picture	**Text | Graphic**

9.1.1.2.1 Change Column Style

The **Column Style Manager** designed to help you managing and designing the selected **Report Table Column** control style. It contains the following properties:

- **Column Header Style Properties:**
  - **Font Family:** Specify the text font family [Arial, Serif…etc.] for the selected report table column's header.
  - **BG Color:** Set the background Color for the selected report table column.
  - **Text Color:** Specify the font color for the selected report table column's header.
  - **Text Size:** Set the font size for the selected report table column's header. Accepts numeric value only.
  - **Text Align:** Align the text for the selected report table column's header to left/ center/ right.
  - **Vertical Align:** Align the text for the selected report table column's header vertically to Middle/Top/Bottom.
  - **Bold:** Set the text to bold for the selected report table column's header.
  - **Italic:** Set the text to Italics for the selected report table column's header.
  - **Underline:** Set the text to Underline for the selected report table column's header.



- **Column Cells Style Properties:**
  - **Font Family:** Specify the text font family [Arial, Serif…etc.] for the selected report table control column's cells.
  - **BG Color:** Set the background Color for the selected report table control column's cells.
  - **Text Color:** Specify the font color for the selected report table control column's cells.
  - **Text Align:** Align the text for the selected report table column's cells to left/ center/ right.
  - **Vertical Align:** Align the text for the selected report table column's cells vertically to Middle/Top/Bottom.
  - **Bold:** Set the text to bold for the selected report table column's cells.
  - **Italic:** Set the text to Italics for the selected report table column's cells.
  - **Underline:** Set the text to Underline for the selected report table column's cells.
- **Column Footer Style Properties:**
  - **Font Family:** Specify the text font family [Arial, Serif…etc.] for the selected report table control column's footer.
  - **BG Color:** Set the background Color for the selected report table control column's footer.

- **Text Color:** Specify the font color for the selected report table control column's footer.
- **Text Align:** Align the text for the selected report table column's footer to left/ center/ right.
- **Vertical Align:** Align the text for the selected report table column's footer vertically to Middle/Top/Bottom.
- **Bold:** Set the text to bold for the selected report table column's footer.
- **Italic:** Set the text to Italics for the selected report table column's footer.
- **Underline:** Set the text to Underline for the selected report table column's footer.

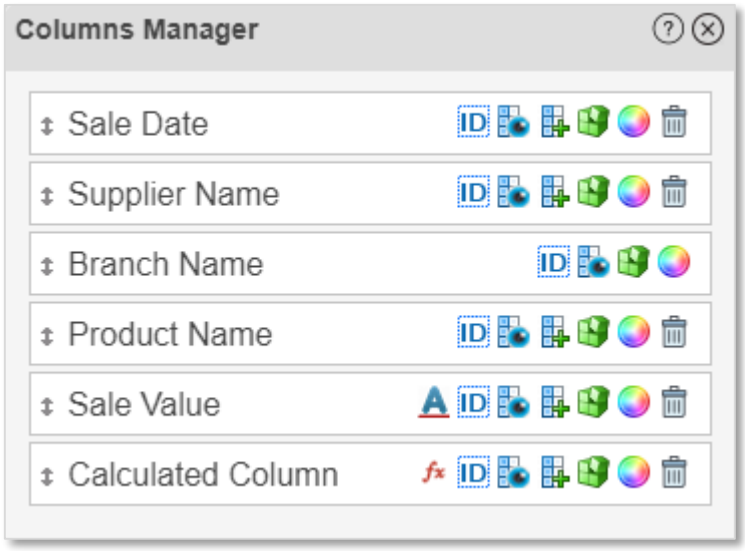
Note: You can click on this icon to remove the color properties for the control (it will become transparent).

9.1.1.2.2 Calculated Column

The **Calculated Column** is a special column's type, which is not related directly to query columns, instead it gets its calculated value from a user defined formula in which the user can references multiple **Report Table** control columns data and returns a specific calculated values in its column rows, by executing mathematical and none-mathematical operations on referenced columns. The **Calculated Column** will have a special button in the **Report Columns Manager** to allow defining formulas for this type of columns. Refer to **Calculated Column Formula** section for more details.

9.1.1.2.3 Calculated Column Formula (*fx*)

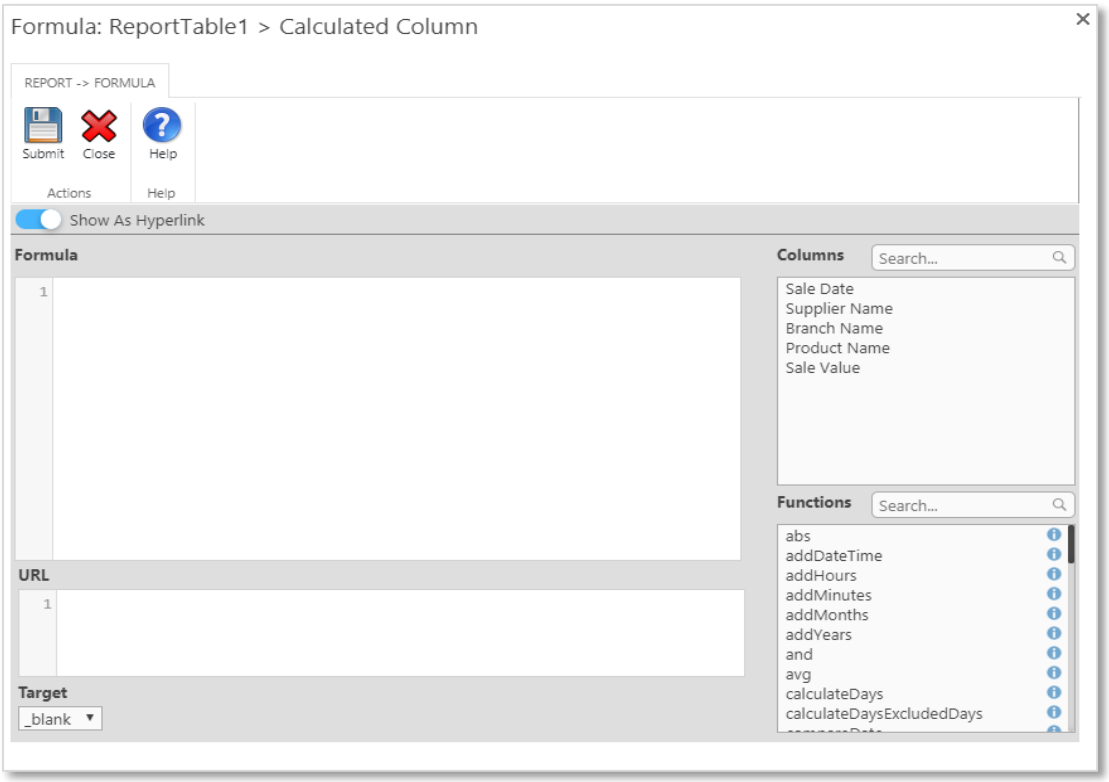
The designer can specify a special type of formulas to a **Calculated Column** type to be able to include calculated/combined and scripted values in the report table columns.



Note that formulas can be applied on all Report Table columns only and on Calculated Columns that were created before them (calculated columns that appear to the left side of the current calculated column you want to apply the formula on", so keep in mind that calculated columns order is significant for running formulas in the report design.

To create a formula for a **Calculated Column**, you need to click on the button on the **Report Columns Manager**.

Once the designer clicks on the button, it will open the **Calculated Column Formula** dialog, which consist of the following parts:



- **Show As Hyperlink:** Switch this option **ON** if you want to have a hyperlink calculated column. In this case, the **URL** and **Target** fields appears in the Formula dialog.

- **Formula:** This input is to have the formula script inside it; the script could consists of a ready built-in functions, JQ or JS code. In case the "**Show As Hyperlink**" switch is **ON**, the **Formula field** will be the URL display text in the report table's calculated column part.
  Example: (col_Sale_x0020_Value1 - col_Discount_x0020_Value1) * 0.9

- **URL:** Set the URL of the formula link. This URL can be a static or a dynamic one, and will appear if the "**Show As Hyperlink**" option is switched **ON**.

  Example: Dynamic link to get the URL of each list item in the report.

  "http://sp2016.sparknit.com/Lists/DateTimesList/DispForm.aspx?ID=" + col_ID_Value1

- **Target:** An attribute that specifies where to open the link. This property can be one of the following options:
  - _blank: Opens the link in a new window or tab.
  - _parent: Opens the link in the parent frame.
  - _self: Opens the link in the same frame.
  - _top: Opens the link in the full body of the window.
  - _search: Opens the link in the browser's Search pane.

- **Columns:** A list includes the **Report Table** available Columns. The designer can search for any column and can include it in the **Formula** or the **URL** fields by double clicking on that column.

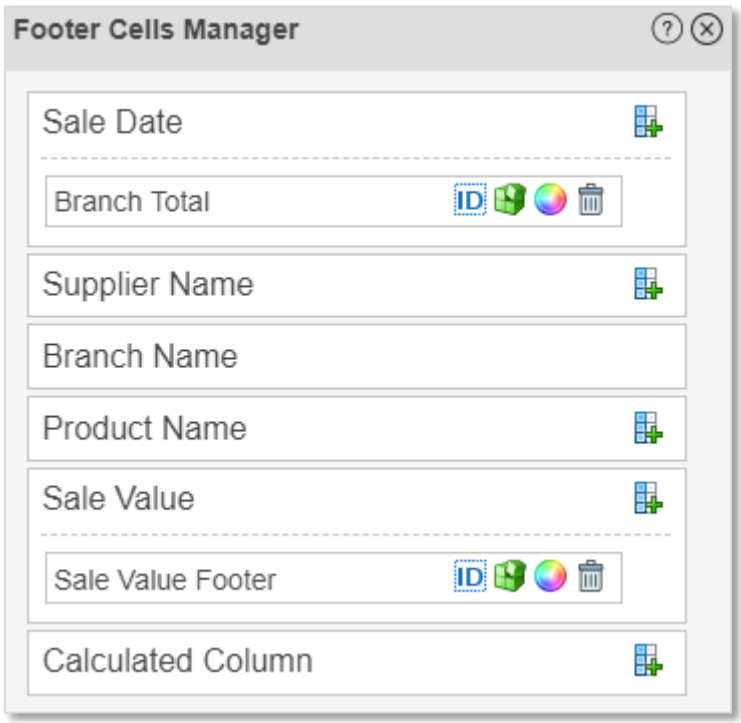
- **Functions:** A list of available built-in functions especially developed for the **Calculated Column Formulas**. The designer can search for any function and can include it in the **Formula** or the **URL** fields by double clicking on it. To show the description of the function, click on the **Functions Description** button next to the name of the function. Note that **Functions Description** shows the description of each function selected in the functions list, which includes the following information: (Format, Description, Function Variables, Arguments Type, Returns and Examples). Refer to the **Calculated Columns Function** section for more details about these functions and their descriptions.

Notes:

- We recommend using the **Columns List** to get the include columns internal names in the formula/URL sections, as their internal names are complex.
- The **Formula Rule** must return a value to be set in the **Calculated Column**; otherwise, it will be considered as an invalid formula and will return an error.

9.1.1.3 Footer Cells Manager

The **Footer Cells Manager** designed to help you managing and designing the selected **Report Table** control's footer cells. Note that creating footer cells will depend on the **Report Table** columns in order to maintain the proportional sizes for columns and cells. The **Footer Cells Manager** contains the following parts:



- **Column Section:** The container of each column in the Report Table control.

Clicking the **Add Footer Cell** button will insert a new **Footer Cell** to the Report Table design beneath the column you clicked this button on, also it will add a new **Footer Cell Section** in the **Column Section** under the column you clicked this button on. Refer to the **Footer Cell** section for more details.

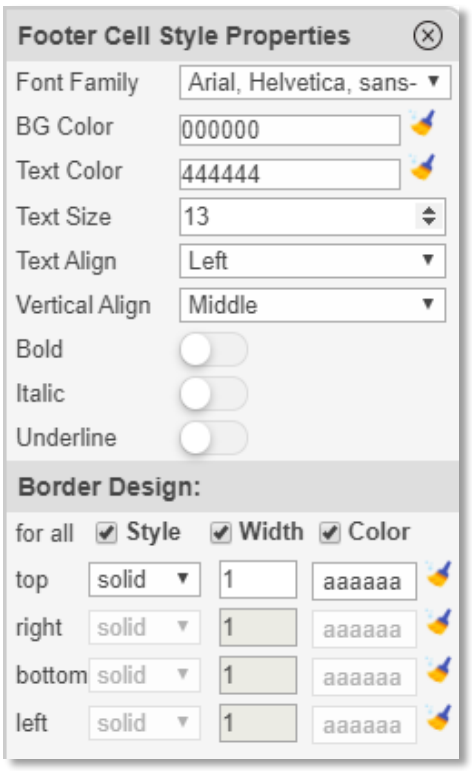
- **Footer Cell Section:** The container of each **Footer Cell** in the Report Table control's column.

You can click on the **Footer Cell** caption to change it the way you want and it will be reflected directly on the selected Report Table footer cell.

- **Delete Footer Cell Button:** Clicking this button will delete the footer cell. A confirmation message will appear before proceeding with the process.
- **Change Cell Style Button:** Clicking this button will open the **Footer Cell Style Manager** dialog. Refer to the **Footer Cell Style Manager** section for more details.
- **Manage Footer Cell Rules Button:** Clicking this button will open **the Footer Cell Rules Manager** dialog. Refer to the **Rules** section for more details.
- **Edit Footer Cell ID:** Clicking this button will allow editing the internal **Footer Cell** ID, which is used in rules and formulas. Once you change the ID, click on the button to apply the change.

9.1.1.3.1 Footer Cell Style

The **Footer Cell Style Manager** designed to help you managing and designing the selected **Report Table Footer Cell** control style. It contains the following properties:



- **Footer Cell Style Properties:**
  - **Font Family:** Specify the text font family [Arial, Serif…etc.] for the selected report table footer cell.
  - **BG Color:** Set the background Color for the selected report table footer's cell.
  - **Text Color:** Specify the font color for the selected report table footer's cell.
  - **Text Size:** Set the font size for the selected report table footer's cell. Accepts numeric value only.
  - **Text Align:** Align the text for the selected report table footer's cell to left/ center/ right.
  - **Bold:** Set the text to bold for the selected report table footer's cell.
  - **Italic:** Set the text to Italics for the selected report table footer's cell.
  - **Underline:** Set the text to Underline for the selected report table footer's cell.
- **Border Design:** To design the border of the selected control. Have the following properties:
  - **for all:** When you check this option, you will control all the four sides of the control borders with one change on the top border properties.
  - **Style:** Set border style for the selected report table control as solid/ dashed/ dotted/ ...
  - **Width:** Set the border width for the selected report table control.
  - **Color:** Set the border color for the selected report table control.

Note: You can click on this icon to remove the color properties for the control (it will become transparent).

9.1.1.3.2 Footer Cell

The **Footer Cell** is a special part of the Report Table control. It's not related directly to report's query columns, instead it gets its value from a user defined **Formula Rules** or **Action Rules** in which the user can references multiple **Report Table** control's columns data and returns a specific calculated values in its cell, by executing mathematical and none-mathematical operations on referenced columns.
The **Footer Cells** deign will depend on the columns and table design and width, in order to maintain the table cells proportional design. In some cases, you may need to create couple of empty cells to keep a strong and logical design of the table.

See also

Reference
addAttribute() function
getAttributeValue() function
modifyAttributeValue() function

9.1.2 Label Control

The **Label** control displays text on the report. It is usually a static control but can be a dynamic one too. A label usually used to for titles, descriptions or highlights.

Control Properties:

- **Control Type:** The type of the selected control.
- **ID:** The Identity of the control. The ID is a unique value, no duplicate IDs allowed in the report.
- **Text:** Set a default value for the control display text. The value will only be used if no value has been specified in the control rules**.** You can click on this icon (Control Text Editor) to add multi-line rich text.
- **Tooltip:** Appears on running the report and the user moves the mouse pointer over this control.
- **CSS Classes:** The HTML class attribute makes it possible to define equal styles for elements with the same class name.
- **Style:** The HTML style code in a free style editor that can be modified on the spot for the selected control.
- **Show Control:** A dropdown list defines how the designer wants this control to be displayed along report's pages if the report "**Details Page**" generated multiple pages, the designer can choose to have this control **On Every Page, On First Page and On Last Page**. Note that this property will only be visible if the control placed on the **Details Page**, and will operate on generated pages of this report's section only.
- **Hidden:** Specifies if the control will be visible or not when running the report.

See also

Reference
clear() function
copyFromTo() function

isEmpty() function
isEqual() function
isGreaterEqualThan() function
isGreaterThan() function
isNotEmpty() function
isNotEqual() function
isNumber() function
isSmallerEqualThan() function
isSmallerThan() function
length() function
math() function
not() function
notContain() function
notEndWith() function
notStartWith() function
getValue() function
setValue() function
max() function
min() function
setFocus() function
split() function
startWith() function
substring() function
sum() function
toLower() function
toUpper() function
toUpperFirst() function
Trim() function
addAttribute() function
getAttributeValue() function
modifyAttributeValue() function
show() function

9.1.3 HTML Text Control

The **HTML Text** control accepts html and CSS scripts as an input and renders these scripts directly in the report. This allows report designers to provide an advanced data presentation for users from within the report without the need to inject these scripts into the report's source or use formatting rules to do that.

Control Properties:

- **Control Type:** The type of the selected control.
- **ID:** The Identity of the control. The ID is a unique value, no duplicate IDs allowed in the report.
- **Text:** The text/script of the HTML/CSS that the control will render. You can click on this icon (Rich Text Editor) to add rich HTML contents using the rich HTML Editor features.
- **Tooltip:** Appears on running the report and the user moves the mouse pointer over this control.
- **CSS Classes:** The HTML class attribute makes it possible to define equal styles for elements with the same class name.
- **Style:** The HTML CSS code. The user can edit this code in the editor and change the control style in the designer and art runtime.
- **Show Control:** A dropdown list defines how the designer wants this control to be displayed along report's pages if the report "**Details Page**" generated multiple pages, the designer can choose to have this control **On Every Page, On First Page and On Last Page**. Note that this property will only be visible if the control placed on the **Details Page**, and will operate on generated pages of this report's section only.
- **Hidden:** Specifies if the control will be visible or not when running the report.

See also

Reference
clear() function
copyFromTo() function
isEmpty() function
isEqual() function
isGreaterEqualThan() function
isGreaterThan() function
isNotEmpty() function
isNotEqual() function
isNumber() function
isSmallerEqualThan() function
isSmallerThan() function
length() function
math() function
not() function
notContain() function
notEndWith() function
notStartWith() function
getValue() function
setValue() function
max() function
min() function
setFocus() function
split() function
startWith() function
substring() function
sum() function
toLower() function
toUpper() function
toUpperFirst() function
Trim() function
addAttribute() function
getAttributeValue() function
modifyAttributeValue() function
show() function
appendRichTextEditor() function

9.1.4 Horizontal Line control

The **Horizontal Line** control defines a graphical horizontal line in the report's design. For example, it can be used to separate contents in the report.

Control Properties:

- **Control Type:** The type of the selected control.
- **ID:** The Identity of the control. The ID is a unique value, no duplicate IDs allowed in the report.
- **Tooltip:** Appears on running the report and the user moves the mouse pointer over this control.
- **CSS Classes:** The HTML class attribute makes it possible to define equal styles for elements with the same class name.
- **Style:** The HTML CSS code. The user can edit this code in the editor and change the control style in the designer and art runtime.
- **Show Control:** A dropdown list defines how the designer wants this control to be displayed along report's pages if the report "**Details Page**" generated multiple pages, the designer can choose to have this control **On Every Page, On First Page and On Last Page**. Note that this property will only be visible if the control placed on the **Details Page**, and will operate on generated pages of this report's section only.

- **Hidden:** Specifies if the control will be visible or not when running the report.

See also

Reference
addAttribute() function
getAttributeValue() function
modifyAttributeValue() function

9.1.5 Vertical Line control

The **Vertical Line** control defines a graphical vertical line in the report's design. For example, it can be used to separate contents in the report.

Control Properties:

- **Control Type:** The type of the selected control.
- **ID:** The Identity of the control. The ID is a unique value, no duplicate IDs allowed in the report.
- **Tooltip:** Appears on running the report and the user moves the mouse pointer over this control.
- **CSS Classes:** The HTML class attribute makes it possible to define equal styles for elements with the same class name.
- **Style:** The HTML CSS code. The user can edit this code in the editor and change the control style in the designer and art runtime.
- **Show Control:** A dropdown list defines how the designer wants this control to be displayed along report's pages if the report "**Details Page**" generated multiple pages, the designer can choose to have this control **On Every Page, On First Page and On Last Page**. Note that this property will only be visible if the control placed on the **Details Page**, and will operate on generated pages of this report's section only.
- **Hidden:** Specifies if the control will be visible or not when running the report.

See also

Reference
addAttribute() function
getAttributeValue() function
modifyAttributeValue() function

9.1.6 Hyperlink Control

The **Hyperlink** control can be used to enter a hyperlink URL and display text in the report design.

Control Properties:

- **Control Type:** The type of the selected control.
  - **ID:** The Identity of the control. The ID is a unique value, no duplicate IDs allowed in the report.
- **Default value:** Set a default display text for the hyperlink. This value will only be used if no value has been specified in the control rules**.**
  - **Tooltip:** Appears on running the report and the user moves the mouse pointer over this control.
- **Url:** Set the hyperlink URL value for the control. If a # is specified (the default), the current opened URL will be opened in a new tab.
- **Target:** An attribute that specifies where to open the linked item. This property can be one of the following options:
  - _blank: Opens the link in a new window or tab.
  - _parent: Opens the link in the parent frame.
  - _self: Opens the link in the same frame.

  - _top: Opens the link in the full body of the window.
  - _search: Opens the link in the browser's Search pane.
- **Edit HyperLink:** Specify if the user will be able to edit this control at runtime, if this property is switched **ON**, an icon will display next to the control, which will show a dialog of the link text and the URL values to be edited by the user at runtime.
- **CSS Classes:** The HTML class attribute makes it possible to define equal styles for elements with the same class name.
- **Style:** The HTML CSS code. The user can edit this code in the editor and change the control style in the designer and art runtime.
- **Show Control:** A dropdown list defines how the designer wants this control to be displayed along report's pages if the report "**Details Page**" generated multiple pages, the designer can choose to have this control **On Every Page, On First Page and On Last Page**. Note that this property will only be visible if the control placed on the **Details Page**, and will operate on generated pages of this report's section only.
- **Disabled:** Specifies if the control will be disabled or not when running the report.
- **ReadOnly:** Specifies if the control will be read only or not when running the report.
- **Hidden:** Specifies if the control will be visible or not when running the report.

See also

Reference
clear() function
setHyperLink() function
getValue() function
setFocus() function
addAttribute() function
getAttributeValue() function
modifyAttributeValue() function

9.1.7 Page Viewer Control

The **Page Viewer** control can be used to display an external page contents in the form. The page viewer control works as an **IFrame** and allows a view of a page or document to be included within the report's design.

Control Properties:

- **Control Type:** The type of the selected control.
- **ID:** The Identity of the control. The ID is a unique value, no duplicate IDs allowed in the report.
- **Default value:** Sets the initial page URL value for the control. The value will only be used if no value has been specified in the control rules**.**
- **Tooltip:** Appears on running the report and the user moves the mouse pointer over this control.
- **CSS Classes:** The HTML class attribute makes it possible to define equal styles for elements with the same class name.
- **Style:** The HTML CSS code. The user can edit this code in the editor and change the control style in the designer and art runtime.
- **Show Control:** A dropdown list defines how the designer wants this control to be displayed along report's pages if the report "**Details Page**" generated multiple pages, the designer can choose to have this control **On Every Page, On First Page and On Last Page**. Note that this property will only be visible if the control placed on the **Details Page**, and will operate on generated pages of this report's section only.
- **Hidden:** Specifies if the control will be visible or not when running the report.

See also

Reference
clear() function

setPageViewer() function
getValue() function
addAttribute() function
getAttributeValue() function
modifyAttributeValue() function

9.1.8 Image Control

The **Image** control is used to display an image on the report.

Control Properties:

- **Control Type:** The type of the selected control.
- **ID:** The Identity of the control. The ID is a unique value, no duplicate IDs allowed in the report.
- **Tooltip:** Appears on running the report and the user moves the mouse pointer over this control.
- **Image URL:** Set the source URL of the image.
- **CSS Classes:** The HTML class attribute makes it possible to define equal styles for elements with the same class name.
- **Style:** The HTML CSS code. The user can edit this code in the editor and change the control style in the designer and art runtime.
- **Show Control:** A dropdown list defines how the designer wants this control to be displayed along report's pages if the report "**Details Page**" generated multiple pages, the designer can choose to have this control **On Every Page, On First Page and On Last Page**. Note that this property will only be visible if the control placed on the **Details Page**, and will operate on generated pages of this report's section only.
- **Hidden:** Specifies if the control will be visible or not when running the report.

See also

Reference
clear() function
setImage() function
getValue() function
addAttribute() function
getAttributeValue() function
modifyAttributeValue() function

9.2 Expression Controls

9.2.1 Sum Control

The **Sum control** can be used to get the sum of a column in the data source and present it in the report design. Note that this control must only be mapped to a numeric type data source column.

Control Properties:

- **Control Type:** The type of the selected control.
- **ID:** The Identity of the control. The ID is a unique value, no duplicate IDs allowed in the report.
- **Text:** Set a display text for the control, you can click on this icon (Control Text Editor) to add multi-line rich text.
- **Tooltip:** Appears on running the report and the user moves the mouse pointer over this control.
- **CSS Classes:** The HTML class attribute makes it possible to define equal styles for elements with the same class name.
- **Style:** The HTML CSS code. The user can edit this code in the editor and change the control style in the designer and art runtime.
- **Decimals:** A numeric value represents the number of decimal places that the control will display.
- **Data Source Column:** A dropdown list contains query's columns. The designer can specify a column to perform the sum operation on its all returned values in report's rows.
- **Show Control:** A dropdown list defines how the designer wants this control to be displayed along report's pages if the report "**Details Page**" generated multiple pages, the designer can choose to have this control **On Every Page, On First Page and On Last Page**. Note that this property will only be visible if the control placed on the **Details Page**, and will operate on generated pages of this report's section only.
- **Hidden:** Specifies if the control will be visible or not when running the report.

See also

Reference
clear() function
setValue() function
addAttribute() function
getAttributeValue() function
modifyAttributeValue() function

9.2.2 Avg Control

The **Avg Control** can be used to get the average value of a column in the data source and present it in the report design. Note that this control must only be mapped to a numeric type data source column.

Control Properties:

- **Control Type:** The type of the selected control.
- **ID:** The Identity of the control. The ID is a unique value, no duplicate IDs allowed in the report.
- **Text:** Set a display text for the control, you can click on this icon (Control Text Editor) to add multi-line rich text.

- **Tooltip:** Appears on running the report and the user moves the mouse pointer over this control.
- **CSS Classes:** The HTML class attribute makes it possible to define equal styles for elements with the same class name.
- **Style:** The HTML CSS code. The user can edit this code in the editor and change the control style in the designer and art runtime.
- **Decimals:** A numeric value represents the number of decimal places that the control will display.
- **Data Source Column:** A dropdown list contains query's columns. The designer can specify a column to perform the average operation on its all returned values in report's rows.
- **Show Control:** A dropdown list defines how the designer wants this control to be displayed along report's pages if the report "**Details Page**" generated multiple pages, the designer can choose to have this control **On Every Page, On First Page and On Last Page**. Note that this property will only be visible if the control placed on the **Details Page**, and will operate on generated pages of this report's section only.
- **Hidden:** Specifies if the control will be visible or not when running the report.

See also

Reference
clear() function
setValue() function
addAttribute() function
getAttributeValue() function
modifyAttributeValue() function

9.2.3 Min Control

The **Min Control** can be used to get the minimum value of a column in the data source and present it in the report design. Note that this control must only be mapped to a numeric type data source column.

Control Properties:

- **Control Type:** The type of the selected control.
- **ID:** The Identity of the control. The ID is a unique value, no duplicate IDs allowed in the report.
- **Text:** Set a display text for the control, you can click on this icon (Control Text Editor) to add multi-line rich text.
- **Tooltip:** Appears on running the report and the user moves the mouse pointer over this control.
- **CSS Classes:** The HTML class attribute makes it possible to define equal styles for elements with the same class name.
- **Style:** The HTML CSS code. The user can edit this code in the editor and change the control style in the designer and art runtime.
- **Decimals:** A numeric value represents the number of decimal places that the control will display.
- **Data Source Column:** A dropdown list contains query's columns. The designer can specify a column to perform the min operation on its all returned values in report's rows.
- **Show Control:** A dropdown list defines how the designer wants this control to be displayed along report's pages if the report "**Details Page**" generated multiple pages, the designer can choose to have this control **On Every Page, On First Page and On Last Page**. Note that this property will only be visible if the control

placed on the **Details Page**, and will operate on generated pages of this report's section only.
- **Hidden:** Specifies if the control will be visible or not when running the report.

See also

Reference
clear() function
setValue() function
addAttribute() function
getAttributeValue() function
modifyAttributeValue() function

9.2.4 Max Control

The **Max Control** can be used to get the maximum value of a column in the data source and present it in the report design. Note that this control must only be mapped to a numeric type data source column.

Control Properties:

- **Control Type:** The type of the selected control.
- **ID:** The Identity of the control. The ID is a unique value, no duplicate IDs allowed in the report.
- **Text:** Set a display text for the control, you can click on this icon (Control Text Editor) to add multi-line rich text.
- **Tooltip:** Appears on running the report and the user moves the mouse pointer over this control.
- **CSS Classes:** The HTML class attribute makes it possible to define equal styles for elements with the same class name.
- **Style:** The HTML CSS code. The user can edit this code in the editor and change the control style in the designer and art runtime.
- **Decimals:** A numeric value represents the number of decimal places that the control will display.
- **Data Source Column:** A dropdown list contains query's columns. The designer can specify a column to perform the max operation on its all returned values in report's rows.
- **Show Control:** A dropdown list defines how the designer wants this control to be displayed along report's pages if the report "**Details Page**" generated multiple pages, the designer can choose to have this control **On Every Page, On First Page and On Last Page**. Note that this property will only be visible if the control placed on the **Details Page**, and will operate on generated pages of this report's section only.
- **Hidden:** Specifies if the control will be visible or not when running the report.

See also

Reference
clear() function
setValue() function
addAttribute() function
getAttributeValue() function
modifyAttributeValue() function

9.2.5 Count Control

The **Count Control** can be used to get the rows count number of a column in the data source and present it in the report design.

Control Properties:

- **Control Type:** The type of the selected control.
- **ID:** The Identity of the control. The ID is a unique value, no duplicate IDs allowed in the report.
- **Text:** Set a display text for the control, you can click on this icon (Control Text Editor) to add multi-line rich text.
- **Tooltip:** Appears on running the report and the user moves the mouse pointer over this control.
- **CSS Classes:** The HTML class attribute makes it possible to define equal styles for elements with the same class name.
- **Style:** The HTML CSS code. The user can edit this code in the editor and change the control style in the designer and art runtime.
- **Data Source Column:** A dropdown list contains query's columns. The designer can specify a column to perform the count operation on its all returned values in report's rows.
- **Show Control:** A dropdown list defines how the designer wants this control to be displayed along report's pages if the report "**Details Page**" generated multiple pages, the designer can choose to have this control **On Every Page, On First Page and On Last Page**. Note that this property will only be visible if the control placed on the **Details Page**, and will operate on generated pages of this report's section only.
- **Hidden:** Specifies if the control will be visible or not when running the report.

See also

Reference
clear() function
setValue() function
addAttribute() function
getAttributeValue() function
modifyAttributeValue() function

9.2.6 Formula Control

The **Formula Control** can be used to perform a calculation/formula on a specific column or multiple columns in the data source and present result in the report design. The designer must create a Formula Rule on the control in order get a calculated value. Refer to the **Rules** section for more details on how to create formula rule.

Control Properties:

- **Control Type:** The type of the selected control.
- **ID:** The Identity of the control. The ID is a unique value, no duplicate IDs allowed in the report.
- **Text:** Set a display text for the control, you can click on this icon (Control Text Editor) to add multi-line rich text.
- **Tooltip:** Appears on running the report and the user moves the mouse pointer over this control.
- **CSS Classes:** The HTML class attribute makes it possible to define equal styles for elements with the same class name.
- **Style:** The HTML CSS code. The user can edit this code in the editor and change the control style in the designer and art runtime.

- **Decimals:** A numeric value represents the number of decimal places that the control will display.
- **Show Control:** A dropdown list defines how the designer wants this control to be displayed along report's pages if the report "**Details Page**" generated multiple pages, the designer can choose to have this control **On Every Page, On First Page and On Last Page**. Note that this property will only be visible if the control placed on the **Details Page**, and will operate on generated pages of this report's section only.
- **Hidden:** Specifies if the control will be visible or not when running the report.

See also

Reference
clear() function
setValue() function
addAttribute() function
getAttributeValue() function
modifyAttributeValue() function

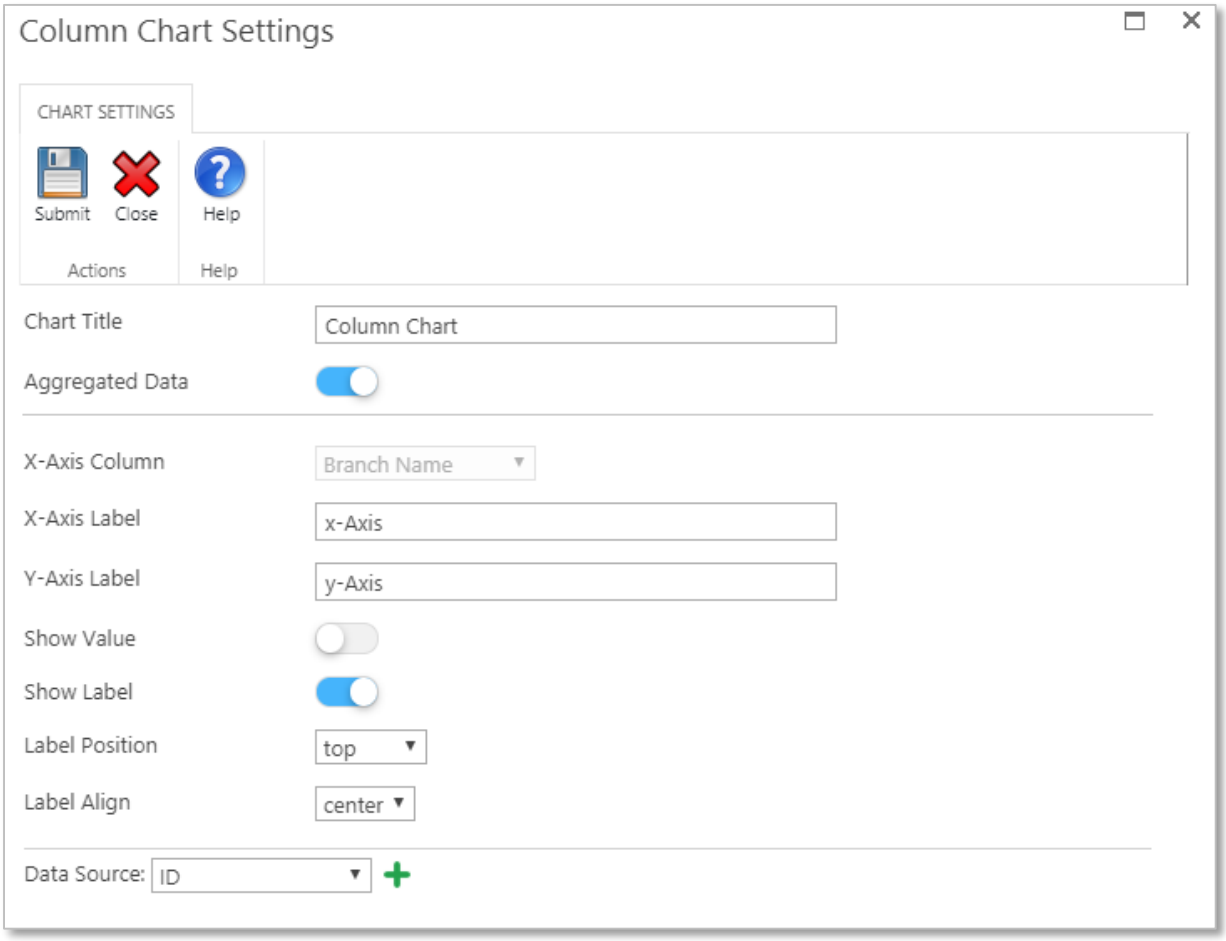
9.3 Chart Controls

9.3.1 Column Chart Control

The **Column Chart** control is a graphic representation of query data; it displays vertical bars going across the chart horizontally, with the values axis being displayed on the left/right side of the chart.

Control Properties:

- **Control Type:** The type of the selected control.
- **ID:** The Identity of the control. The ID is a unique value, no duplicate IDs allowed in the report.
- **Tooltip:** Appears on running the report and the user moves the mouse pointer over this control.
- **CSS Classes:** The HTML class attribute makes it possible to define equal styles for elements with the same class name.
- **Style:** The HTML CSS code. The user can edit this code in the editor and change the control style in the designer and art runtime.
- **Chart Settings:** Clicking this button will display the chart's settings dialog which consist of the following properties:



  - **Chart Title:** Set the chart title, which will appear at the top of the chart.
  - **Aggregated Data:** This property will appear if the query is aggregated "grouped by". If switched **ON** then the data will be presented in the chart based on the "**Group By Column**". Refer to the **Query Builder** section for more information about how to create an aggregated data query.
  - **X-Axis Column:** Specify the X-Axis column to represent its data in the chart's X-Axis. This property will be enabled if the **Aggregated Data**

property is switched OFF or if the report's query is not "grouped by" a column.
    - **X-Axis Label:** Set the X-Axis label text of the chart.
    - **Y-Axis Label:** Set the Y-Axis label text of the chart.
    - **Show Value:** Specify if you want to show data values on the chart.
    - **Show Label:** Specify if you want to show Labels on the chart.
    - **Label Position:** Specify the position of the label (Top, left, bottom and right). This property will only show if the **Show Label** property is switched **ON**.
    - **Label Align:** Specify the alignment of the label (start, center and end). This property will only show if the **Show Label** property is switched **ON**.
    - **Data Source:** Specify the data source column to be represented in the chart. You can add multiple data source columns by clicking the **Add** icon  .
- **Show Control:** A dropdown list defines how the designer wants this control to be displayed along report's pages if the report "**Details Page**" generated multiple pages, the designer can choose to have this control **On Every Page, On First Page and On Last Page**. Note that this property will only be visible if the control placed on the **Details Page**, and will operate on generated pages of this report's section only.
- **Hidden:** Specifies if the control will be visible or not when running the report.

Note: Double click on the control to open the **Control Style Properties** panel and design it the way you want. Refer to the **Control Style Properties** part in the **Report Design Ribbon** section for more details.

See also

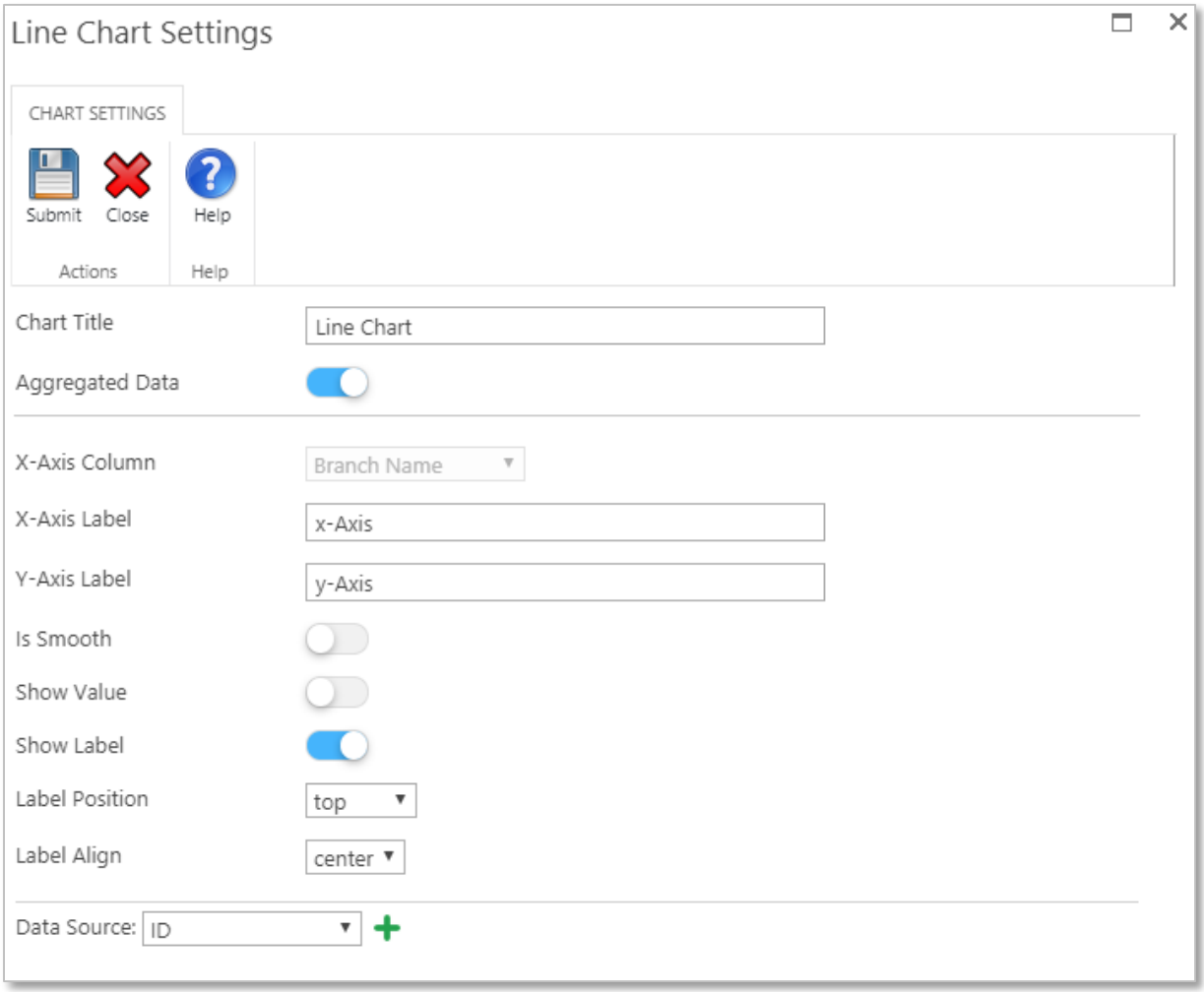
Reference
addAttribute() function
getAttributeValue() function
modifyAttributeValue() function

9.3.2 Line Chart Control

The **Line Chart** control is a type of chart which displays information as a series of data points called 'markers' connected by straight line segments.

Control Properties:

- **Control Type:** The type of the selected control.
- **ID:** The Identity of the control. The ID is a unique value, no duplicate IDs allowed in the report.
- **Tooltip:** Appears on running the report and the user moves the mouse pointer over this control.
- **CSS Classes:** The HTML class attribute makes it possible to define equal styles for elements with the same class name.
- **Style:** The HTML CSS code. The user can edit this code in the editor and change the control style in the designer and art runtime.
- **Chart Settings:** Clicking this button will display the chart's settings dialog which consist of the following properties:



- **Chart Title:** Set the chart title, which will appear at the top of the chart.
- **Aggregated Data:** This property will appear if the query is aggregated "grouped by". If switched **ON** then the data will be presented in the chart based on the "**Group By Column**". Refer to the **Query Builder** section for more information about how to create an aggregated data query.
- **X-Axis Column:** Specify the X-Axis column to represent its data in the chart's X-Axis. This property will be enabled if the **Aggregated Data** property is switched OFF or if the report's query is not "grouped by" a column.
- **X-Axis Label:** Set the X-Axis label text of the chart.
- **Y-Axis Label:** Set the Y-Axis label text of the chart.
- **Is Smooth:** A smoother line is a line that is fitted to the data that helps you explore the potential relationships between two variables without fitting a specific model, such as a regression line or a theoretical distribution. Smoother lines are most useful when the curvature of the relationship does not change sharply.
- **Show Value:** Specify if you want to show data values on the chart.
- **Show Label:** Specify if you want to show Labels on the chart.
- **Label Position:** Specify the position of the label (Top, left, bottom and right). This property will only show if the **Show Label** property is switched **ON**.
- **Label Align:** Specify the alignment of the label (start, center and end). This property will only show if the **Show Label** property is switched **ON**.

- **Data Source:** Specify the data source column to be represented in the chart. You can add multiple data source columns by clicking the **Add** icon  .

- **Show Control:** A dropdown list defines how the designer wants this control to be displayed along report's pages if the report "**Details Page**" generated multiple pages, the designer can choose to have this control **On Every Page, On First Page and On Last Page**. Note that this property will only be visible if the control placed on the **Details Page**, and will operate on generated pages of this report's section only.
- **Hidden:** Specifies if the control will be visible or not when running the report.

Note: Double click on the control to open the **Control Style Properties** panel and design it the way you want. Refer to the **Control Style Properties** part in the **Report Design Ribbon** section for more details.

See also

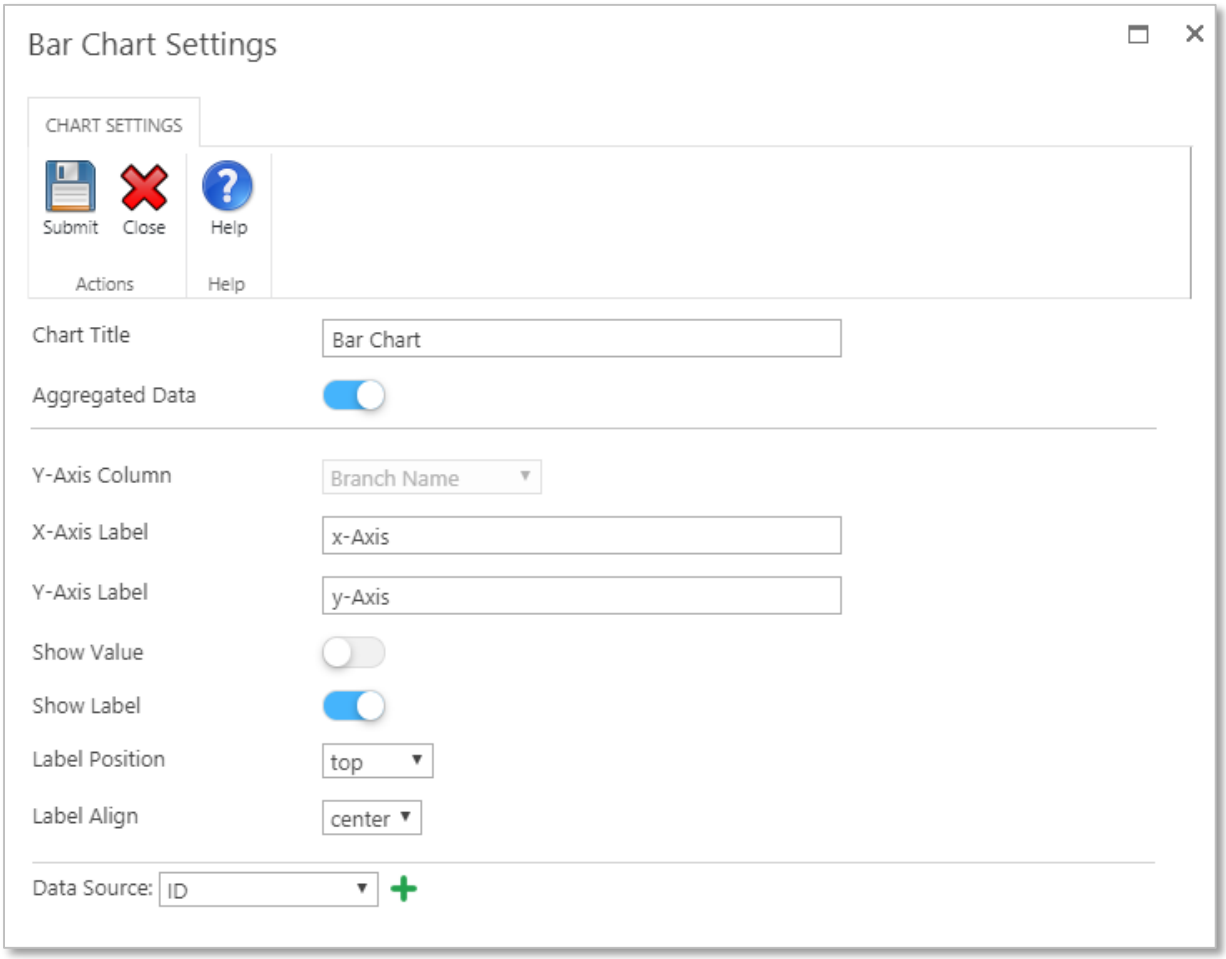
Reference
addAttribute() function
getAttributeValue() function
modifyAttributeValue() function

9.3.3 Bar Chart Control

The **Bar Chart** control presents categorical data with rectangular bars with heights or lengths proportional to the values that they represent.

Control Properties:

- **Control Type:** The type of the selected control.
- **ID:** The Identity of the control. The ID is a unique value, no duplicate IDs allowed in the report.
- **Tooltip:** Appears on running the report and the user moves the mouse pointer over this control.
- **CSS Classes:** The HTML class attribute makes it possible to define equal styles for elements with the same class name.
- **Style:** The HTML CSS code. The user can edit this code in the editor and change the control style in the designer and art runtime.
- **Chart Settings:** Clicking this button will display the chart's settings dialog which consist of the following properties:



- **Chart Title:** Set the chart title, which will appear at the top of the chart.
- **Aggregated Data:** This property will appear if the query is aggregated "grouped by". If switched **ON** then the data will be presented in the chart based on the "**Group By Column**". Refer to the **Query Builder** section for more information about how to create an aggregated data query.
- **X-Axis Column:** Specify the X-Axis column to represent its data in the chart's X-Axis. This property will be enabled if the **Aggregated Data** property is switched OFF or if the report's query is not "grouped by" a column.
- **X-Axis Label:** Set the X-Axis label text of the chart.
- **Y-Axis Label:** Set the Y-Axis label text of the chart.
- **Show Value:** Specify if you want to show data values on the chart.
- **Show Label:** Specify if you want to show Labels on the chart.
- **Label Position:** Specify the position of the label (Top, left, bottom and right). This property will only show if the **Show Label** property is switched **ON**.
- **Label Align:** Specify the alignment of the label (start, center and end). This property will only show if the **Show Label** property is switched **ON**.
- **Data Source:** Specify the data source column to be represented in the chart. You can add multiple data source columns by clicking the **Add** icon  .

- **Show Control:** A dropdown list defines how the designer wants this control to be displayed along report's pages if the report "**Details Page**" generated multiple pages, the designer can choose to have this control **On Every Page, On First Page and On Last Page**. Note that this property will only be visible if the control

placed on the **Details Page**, and will operate on generated pages of this report's section only.
- **Hidden:** Specifies if the control will be visible or not when running the report.

Note: Double click on the control to open the **Control Style Properties** panel and design it the way you want. Refer to the **Control Style Properties** part in the **Report Design Ribbon** section for more details.

See also

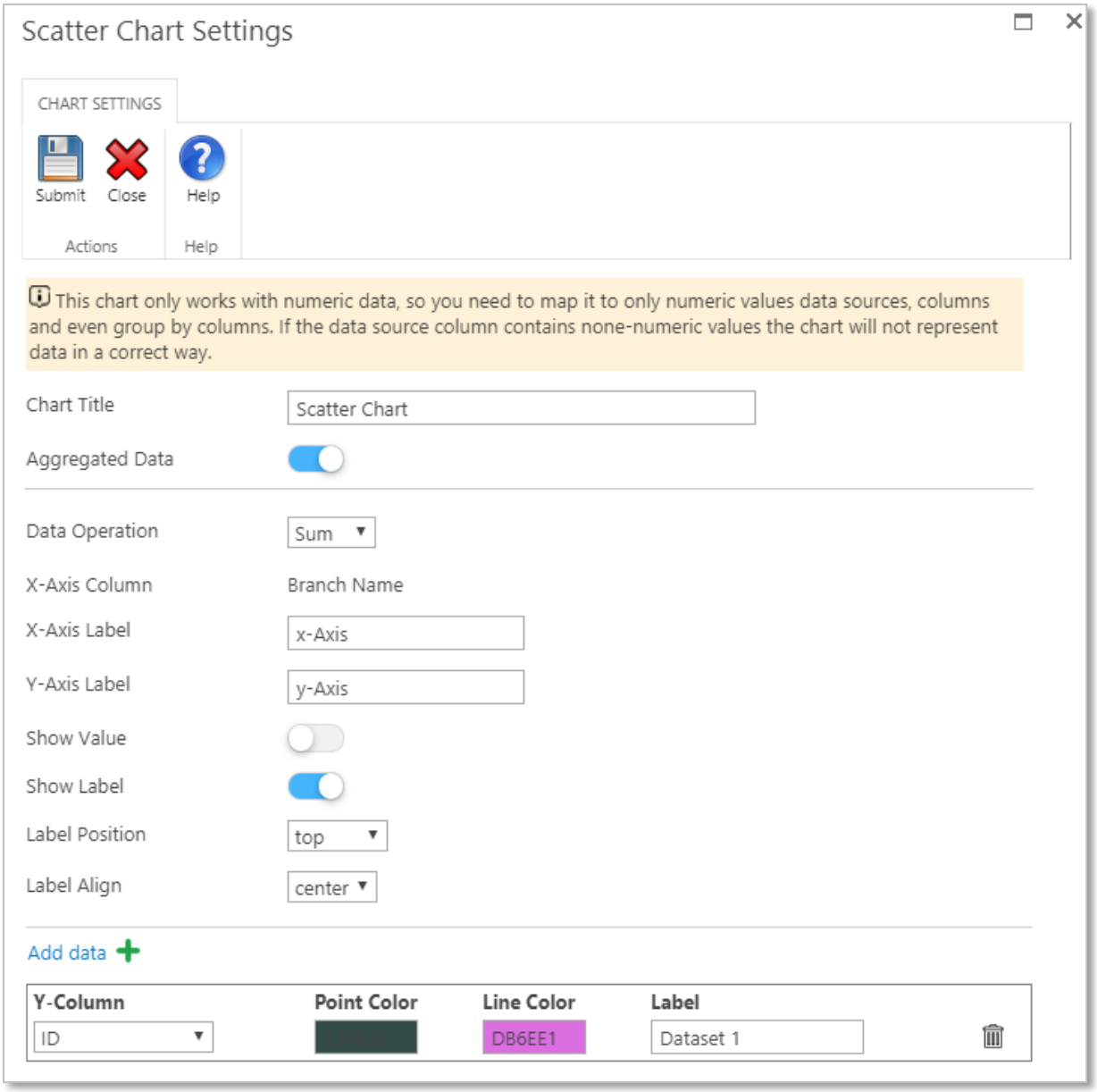
Reference
addAttribute() function
getAttributeValue() function
modifyAttributeValue() function

9.3.4 Scatter Chart Control

The **Scatter Chart** control represents a mathematical diagram using **Cartesian** coordinates to display values for typically two variables for a set of data. If the points are coded (color/shape/size), one additional variable can be displayed. The data are displayed as a collection of points, each having the value of one variable determining the position on the horizontal axis and the value of the other variable determining the position on the vertical axis.

Note: This chart only works with numeric data, so you need to map it to only numeric values data sources, columns and even group by columns. If the data source column contains none-numeric values, the chart will not represent data in a correct way.

Control Properties:
- **Control Type:** The type of the selected control.
- **ID:** The Identity of the control. The ID is a unique value, no duplicate IDs allowed in the report.
- **Tooltip:** Appears on running the report and the user moves the mouse pointer over this control.
- **CSS Classes:** The HTML class attribute makes it possible to define equal styles for elements with the same class name.
- **Style:** The HTML CSS code. The user can edit this code in the editor and change the control style in the designer and art runtime.
- **Chart Settings:** Clicking this button will display the chart's settings dialog which consist of the following properties:



- **Chart Title:** Set the chart title, which will appear at the top of the chart.
- **Aggregated Data:** This property will appear if the query is aggregated "grouped by". If switched **ON** then the data will be presented in the chart based on the "**Group By Column**". Refer to the **Query Builder** section for more information about how to create an aggregated data query.
- **Data Operation:** Specify the mathematical operation that you want the chart to represent data based on (Sum, Avg, Min, Max and Count). This property will be visible if the **Aggregated Data** property is switched ON.
- **X-Axis Column:** Specify the X-Axis column to represent its data in the chart's X-Axis. This property will be visible if the **Aggregated Data** property is switched ON.
- **X-Axis Label:** Set the X-Axis label text of the chart.
- **Y-Axis Label:** Set the Y-Axis label text of the chart.
- **Show Value:** Specify if you want to show data values on the chart.
- **Show Label:** Specify if you want to show Labels on the chart.
- **Label Position:** Specify the position of the label (Top, left, bottom and right). This property will only show if the **Show Label** property is switched **ON**.

- **Label Align:** Specify the alignment of the label (start, center and end). This property will only show if the **Show Label** property is switched **ON**.
- **Add data:** Specify the data source column and its properties (Point Color, Line Color and Label) to be represented in the chart. You can add multiple data source's columns and their properties by clicking the **Add data** icon  .

- **Show Control:** A dropdown list defines how the designer wants this control to be displayed along report's pages if the report "**Details Page**" generated multiple pages, the designer can choose to have this control **On Every Page, On First Page and On Last Page**. Note that this property will only be visible if the control placed on the **Details Page**, and will operate on generated pages of this report's section only.
- **Hidden:** Specifies if the control will be visible or not when running the report.

Note: Double click on the control to open the **Control Style Properties** panel and design it the way you want. Refer to the **Control Style Properties** part in the **Report Design Ribbon** section for more details.

See also

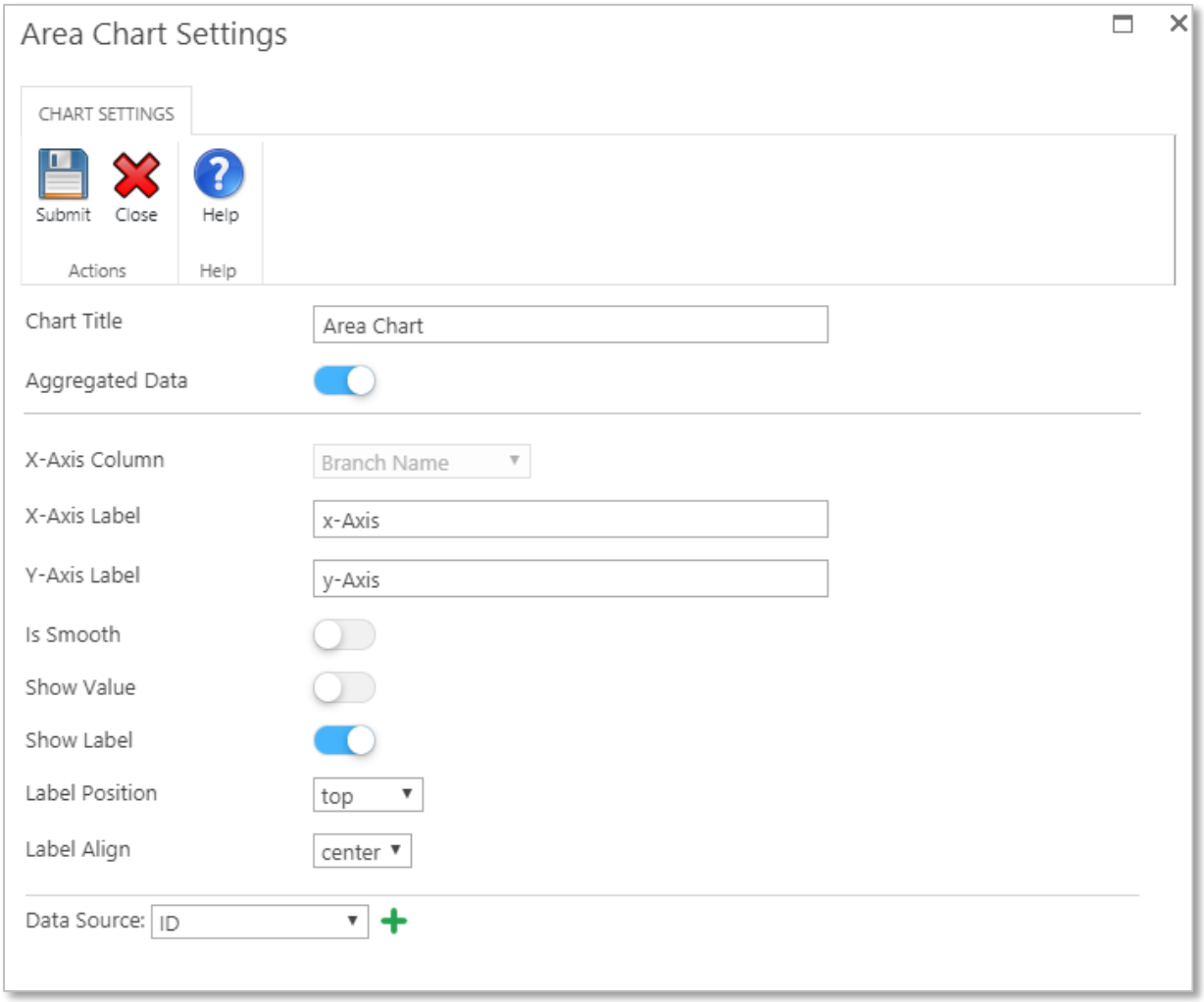
Reference
addAttribute() function
getAttributeValue() function
modifyAttributeValue() function

9.3.5 Area Chart Control

The **Area Chart** control displays graphically quantitative data. It is based on the line chart. The area between axis and line are commonly emphasized with colors, textures and hatchings. Commonly one compares two or more quantities with an area chart.

Control Properties:

- **Control Type:** The type of the selected control.
- **ID:** The Identity of the control. The ID is a unique value, no duplicate IDs allowed in the report.
- **Tooltip:** Appears on running the report and the user moves the mouse pointer over this control.
- **CSS Classes:** The HTML class attribute makes it possible to define equal styles for elements with the same class name.
- **Style:** The HTML CSS code. The user can edit this code in the editor and change the control style in the designer and art runtime.
- **Chart Settings:** Clicking this button will display the chart's settings dialog which consist of the following properties:



- **Chart Title:** Set the chart title, which will appear at the top of the chart.
- **Aggregated Data:** This property will appear if the query is aggregated "grouped by". If switched **ON** then the data will be presented in the chart based on the "**Group By Column**". Refer to the **Query Builder** section for more information about how to create an aggregated data query.
- **X-Axis Column:** Specify the X-Axis column to represent its data in the chart's X-Axis. This property will be enabled if the **Aggregated Data** property is switched OFF or if the report's query is not "grouped by" a column.
- **X-Axis Label:** Set the X-Axis label text of the chart.
- **Y-Axis Label:** Set the Y-Axis label text of the chart.
- **Is Smooth:** A smoother line is a line that is fitted to the data that helps you explore the potential relationships between two variables without fitting a specific model, such as a regression line or a theoretical distribution. Smoother lines are most useful when the curvature of the relationship does not change sharply.
- **Show Value:** Specify if you want to show data values on the chart.
- **Show Label:** Specify if you want to show Labels on the chart.
- **Label Position:** Specify the position of the label (Top, left, bottom and right). This property will only show if the **Show Label** property is switched **ON**.
- **Label Align:** Specify the alignment of the label (start, center and end). This property will only show if the **Show Label** property is switched **ON**.

- **Data Source:** Specify the data source column to be represented in the chart. You can add multiple data source columns by clicking the **Add** icon  .

- **Show Control:** A dropdown list defines how the designer wants this control to be displayed along report's pages if the report "**Details Page**" generated multiple pages, the designer can choose to have this control **On Every Page, On First Page and On Last Page**. Note that this property will only be visible if the control placed on the **Details Page**, and will operate on generated pages of this report's section only.
- **Hidden:** Specifies if the control will be visible or not when running the report.

Note: Double click on the control to open the **Control Style Properties** panel and design it the way you want. Refer to the **Control Style Properties** part in the **Report Design Ribbon** section for more details.

See also

Reference
addAttribute() function
getAttributeValue() function
modifyAttributeValue() function

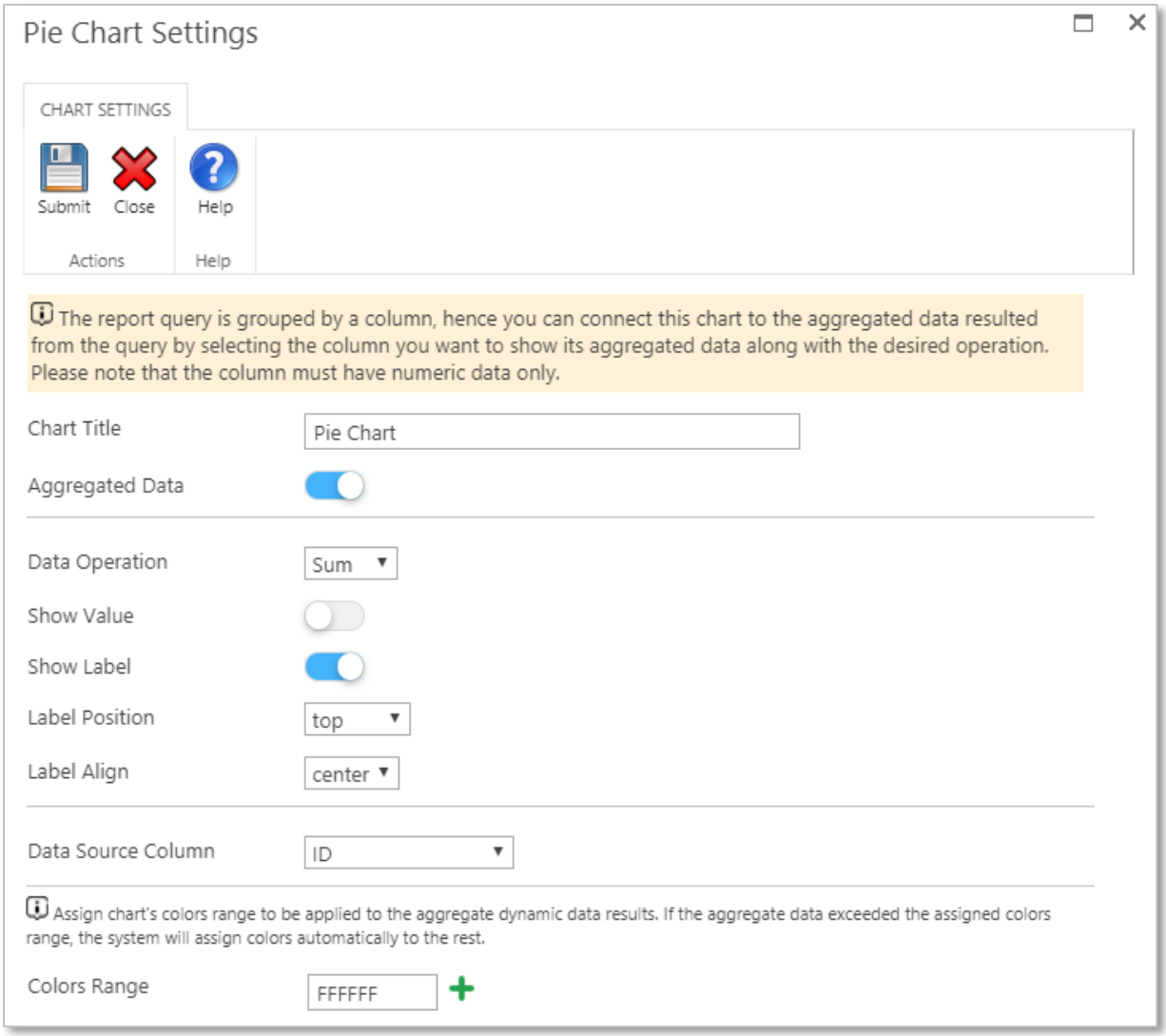
9.3.6 Pie Chart Control

The **Pie Chart** control is a circular statistical graphic, which is divided into slices to illustrate numerical proportion. In a pie chart, the arc length of each slice is proportional to the quantity it represents.

Note: If the report's query is "Grouped By" a column, you can connect this control to the aggregated data resulted from the query by selecting the column you want to show its aggregated data along with the desired operation. Please note that the column must have numeric data only.

Control Properties:

- **Control Type:** The type of the selected control.
- **ID:** The Identity of the control. The ID is a unique value, no duplicate IDs allowed in the report.
- **Tooltip:** Appears on running the report and the user moves the mouse pointer over this control.
- **CSS Classes:** The HTML class attribute makes it possible to define equal styles for elements with the same class name.
- **Style:** The HTML CSS code. The user can edit this code in the editor and change the control style in the designer and art runtime.
- **Chart Settings:** Clicking this button will display the chart's settings dialog which consist of the following properties:



- **Chart Title:** Set the chart title, which will appear at the top of the chart.
- **Aggregated Data:** This property will appear if the query is aggregated "grouped by". If switched **ON** then the data will be presented in the chart based on the "**Group By Column**". Refer to the **Query Builder** section for more information about how to create an aggregated data query.
- **Data Operation:** Specify the mathematical operation that you want the chart to represent data based on (Sum, Avg, Min, Max and Count). This property will be visible if the **Aggregated Data** property is switched ON.
- **Show Value:** Specify if you want to show data values on the chart.
- **Show Label:** Specify if you want to show Labels on the chart.
- **Label Position:** Specify the position of the label (Top, left, bottom and right). This property will only show if the **Show Label** property is switched **ON**.
- **Label Align:** Specify the alignment of the label (start, center and end). This property will only show if the **Show Label** property is switched **ON**.
- **Data Source Column:** Specify the data source column and its properties (Background Color and Label) to be represented in the chart. You can add multiple data source's columns and their properties by clicking the **Add data** icon . In case of aggregated query data "Group By", this property will represents a single data source column and will represents the aggregated values automatically on the chart.

  - **Colors Range:** Assign chart's colors range to be applied to the aggregate dynamic data results. If the aggregate data exceeded the assigned colors range, the system will assign colors automatically to the rest. This property will be visible if the **Aggregated Data** property is switched ON.
- **Show Control:** A dropdown list defines how the designer wants this control to be displayed along report's pages if the report "**Details Page**" generated multiple pages, the designer can choose to have this control **On Every Page, On First Page and On Last Page**. Note that this property will only be visible if the control placed on the **Details Page**, and will operate on generated pages of this report's section only.
- **Hidden:** Specifies if the control will be visible or not when running the report.

Note: Double click on the control to open the **Control Style Properties** panel and design it the way you want. Refer to the **Control Style Properties** part in the **Report Design Ribbon** section for more details.

See also

Reference
addAttribute() function
getAttributeValue() function
modifyAttributeValue() function

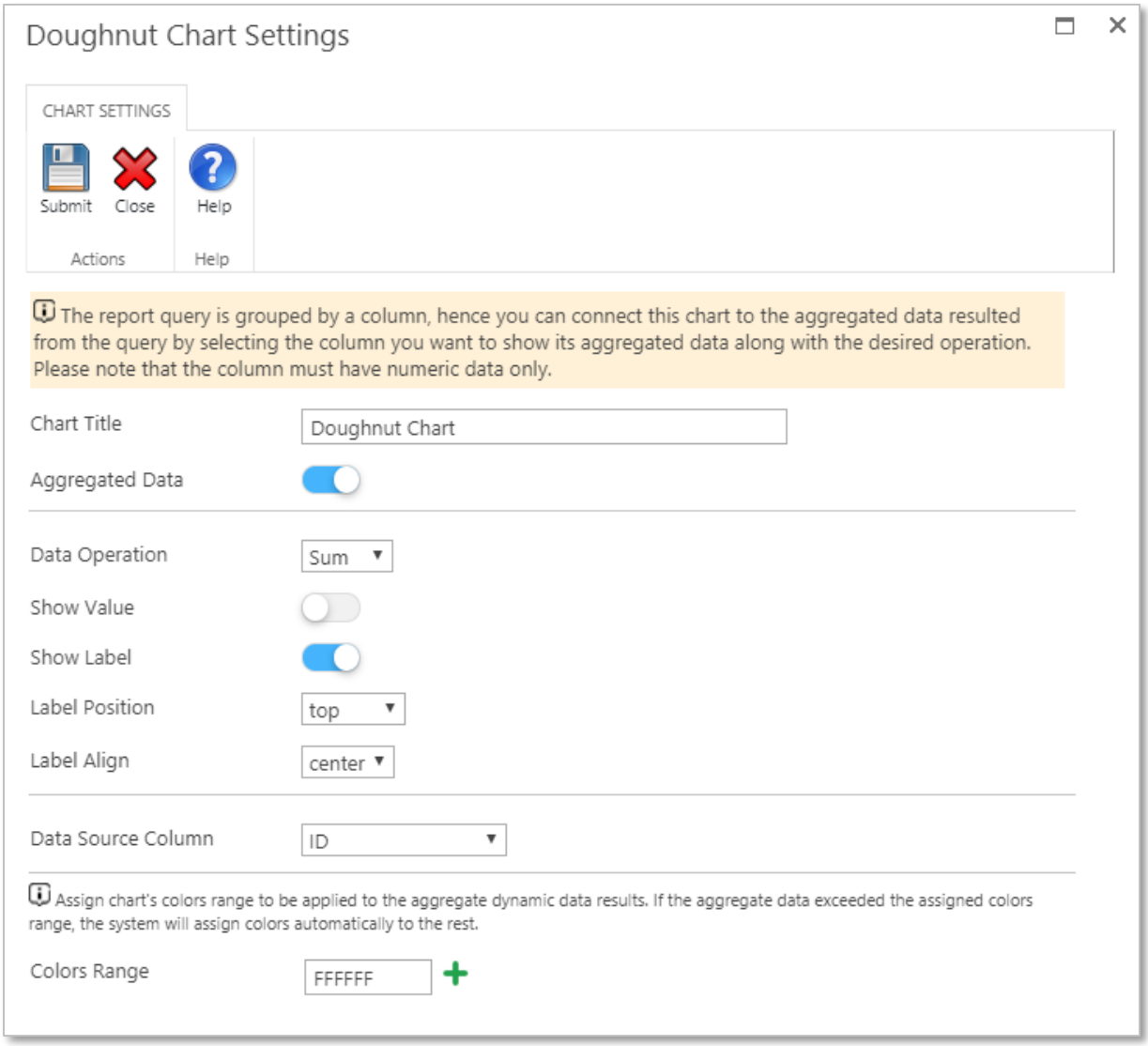
9.3.7 Doughnut Chart Control

The **Doughnut Chart** control is meant to express a "part-to-whole" relationship, where all pieces together represent 100%. Doughnut charts work best to display data with a small number of categories (2-10). For example, you could use a doughnut chart to plot survey questions with a small number of answers, data split by gender, Windows vs. Mac users, or other data where categories are limited.
Doughnut Chart should be avoided when there are many categories, or when categories do not sum to 100%.

Note: If the report's query is "Grouped By" a column, you can connect this control to the aggregated data resulted from the query by selecting the column you want to show its aggregated data along with the desired operation. Please note that the column must have numeric data only.

Control Properties:

- **Control Type:** The type of the selected control.
- **ID:** The Identity of the control. The ID is a unique value, no duplicate IDs allowed in the report.
- **Tooltip:** Appears on running the report and the user moves the mouse pointer over this control.
- **CSS Classes:** The HTML class attribute makes it possible to define equal styles for elements with the same class name.
- **Style:** The HTML CSS code. The user can edit this code in the editor and change the control style in the designer and art runtime.
- **Chart Settings:** Clicking this button will display the chart's settings dialog which consist of the following properties:



- **Chart Title:** Set the chart title, which will appear at the top of the chart.
- **Aggregated Data:** This property will appear if the query is aggregated "grouped by". If switched **ON** then the data will be presented in the chart based on the "**Group By Column**". Refer to the **Query Builder** section for more information about how to create an aggregated data query.
- **Data Operation:** Specify the mathematical operation that you want the chart to represent data based on (Sum, Avg, Min, Max and Count). This property will be visible if the **Aggregated Data** property is switched ON.
- **Show Value:** Specify if you want to show data values on the chart.
- **Show Label:** Specify if you want to show Labels on the chart.
- **Label Position:** Specify the position of the label (Top, left, bottom and right). This property will only show if the **Show Label** property is switched **ON**.
- **Label Align:** Specify the alignment of the label (start, center and end). This property will only show if the **Show Label** property is switched **ON**.
- **Data Source Column:** Specify the data source column and its properties (Background Color and Label) to be represented in the chart. You can add multiple data source's columns and their properties by clicking the **Add data** icon . In case of aggregated query data "Group By", this property will represents a single data source column and will represents the aggregated values automatically on the chart.

  - **Colors Range:** Assign chart's colors range to be applied to the aggregate dynamic data results. If the aggregate data exceeded the assigned colors range, the system will assign colors automatically to the rest. This property will be visible if the **Aggregated Data** property is switched ON.
- **Show Control:** A dropdown list defines how the designer wants this control to be displayed along report's pages if the report "**Details Page**" generated multiple pages, the designer can choose to have this control **On Every Page, On First Page and On Last Page**. Note that this property will only be visible if the control placed on the **Details Page**, and will operate on generated pages of this report's section only.
- **Hidden:** Specifies if the control will be visible or not when running the report.

Note: Double click on the control to open the **Control Style Properties** panel and design it the way you want. Refer to the **Control Style Properties** part in the **Report Design Ribbon** section for more details.

See also

Reference
addAttribute() function
getAttributeValue() function
modifyAttributeValue() function

9.3.8 Radial Gauge Control

The **Radial Gauge** control has a circular arc and shows a single value that measures progress toward a goal or a Key Performance Indicator (KPI). The line (or needle) represents the goal or target value. The shading represents the progress toward that goal while the value inside the arc represents the progress value.

Control Properties:

- **Control Type:** The type of the selected control.
- **ID:** The Identity of the control. The ID is a unique value, no duplicate IDs allowed in the report.
- **Tooltip:** Appears on running the report and the user moves the mouse pointer over this control.
- **CSS Classes:** The HTML class attribute makes it possible to define equal styles for elements with the same class name.
- **Title:** Set the title of the control, the title will appear at the top of the control.
- **Title Color:** Specify the control's **Title** text color.
- **Units:** Set the units of the values i.e.: KM/h.
- **Units Color:** Specify Units color.
- **Min Value:** Set the minimum value of the gauge. Accepts only numeric value.
- **Max Value:** Set the maximum value of the gauge. Accepts only numeric value.
- **Start Angle:** Set the start angle of the gauge. Accepts only numeric value.
- **Ticks Angle:** Set the Ticks (End) angle of the gauge. Accepts only numeric value.
- **Major Ticks:** Set the Ticks range (numeric values separated by commas), i.e.: (0,20,40,60,80,100,120,140,160,180) these values will be shown in the gauge.
- **Minor Ticks:** Set the minor Ticks where the gauge needle will point to when the value is too low. Accepts only numeric value.
- **Major Ticks Color:** Specify **Major Ticks** color.
- **Minor Ticks Color:** Specify **Minor Ticks** color.
- **Numbers Color:** Specify the gauge numbers' color.
- **Show Stroke Ticks:** Switching this property **ON** will display the gauge ticks stroke "ticks arc".
- **Plate Color:** Specify the gauge circular shape color.
- **Show Borders:** Switching this property **ON** will display the gauge border.

- **Border Shadow Width:** Set the gauge border shadow's width. Accepts only numeric value.
- **Needle Type:** Specify the gauge's needle type (Arrow or Line).
- **Needle Width:** Specify the gauge's needle width. Accepts only numeric value.
- **Needle Circle Size:** Specify the gauge needle circle size. Accepts only numeric value.
- **Needle Circle Outer:** Switching this property **ON** will display the gauge's needle central circle.
- **Needle Circle Outer Color:** Specify the gauge's needle circle outer color.
- **Needle Circle Outer End Color:** Specify the gauge's needle circle outer End color.
- **Needle Circle Inner:** Switching this property **ON** will display the gauge's needle central inner circle.
- **Needle Circle Inner Color:** Specify the gauge's needle circle inner color.
- **Needle Circle Inner End Color:** Specify the gauge's needle circle inner end color.
- **Needle Shadow Down Color:** Specify the gauge's needle shadow down color.
- **Animation Duration:** Set the gauge's animation duration value when the needle move to the targeted value. Accepts only numeric value.
- **Gauge Animation Rule:** Specify the gauge animation rule (linear, quad, dequad, quint, dequint, cycle, decycle, bounce, debounce, elastic, delastic).
- **Show Value Box:** Switching this property **ON** will display the gauge's value box inside the gauge's control shape.
- **Value Box Border Radius:** Set the Value Box Border Radius value of the gauge. Accepts only numeric value.
- **Value Box Rect Color:** Specify the Value Box rectangle color.
- **Value Box Rect end Color:** Specify the Value Box rectangle end color.
- **Color Highlights:** Clicking this button will display the gauge's color highlights dialog which consist of the following properties:
  - **From:** Set the highlight "From" value. Accepts only numeric value.
  - **To:** Set the highlight "To" value. Accepts only numeric value.
  - **Color:** Specify the gauge highlight color.
  - **Add New:** You can add multiple highlights by clicking the **Add data** icon  .
- **Style:** The HTML CSS code. The user can edit this code in the editor and change the control style in the designer and art runtime.
- **Operator:** Specify the mathematical operation that you want the gauge control to represent data based on (Sum, Avg, Min, Max and Count).
- **Data Source:** Specify the data source column to be represented in the chart.
- **Show Control:** A dropdown list defines how the designer wants this control to be displayed along report's pages if the report "**Details Page**" generated multiple pages, the designer can choose to have this control **On Every Page, On First Page and On Last Page**. Note that this property will only be visible if the control placed on the **Details Page**, and will operate on generated pages of this report's section only.
- **Hidden:** Specifies if the control will be visible or not when running the report.

Note: Double click on the control to open the **Control Style Properties** panel and design it the way you want. Refer to the **Control Style Properties** part in the **Report Design Ribbon** section for more details.

See also

Reference
addAttribute() function
getAttributeValue() function
modifyAttributeValue() function

9.3.9 Linear Gauge Control

The **Linear Gauge** control is visual representation of a measuring device with a horizontal or vertical scale and a pointer or multiple pointers indicating particular values. **Linear Gauges** can represent thermometers, radio scales, battery indicators, rulers, and any other devices with straight line-shaped scales.

Control Properties:

- **Control Type:** The type of the selected control.
- **ID:** The Identity of the control. The ID is a unique value, no duplicate IDs allowed in the report.
- **Tooltip:** Appears on running the report and the user moves the mouse pointer over this control.
- **CSS Classes:** The HTML class attribute makes it possible to define equal styles for elements with the same class name.
- **Title:** Set the title of the control, the title will appear at the top of the control.
- **Title Color:** Specify the control's **Title** text color.
- **Min Value:** Set the minimum value of the gauge. Accepts only numeric value.
- **Max Value:** Set the maximum value of the gauge. Accepts only numeric value.
- **Major Ticks:** Set the Ticks range (numeric values separated by commas), i.e.: (0,20,40,60,80,100,120,140,160,180) these values will be shown in the gauge.
- **Minor Ticks:** Set the minor Ticks where the gauge needle will point to when the value is too low. Accepts only numeric value.
- **Numbers Color:** Specify the gauge numbers' color.
- **Show Stroke Ticks:** Switching this property **ON** will display the gauge ticks stroke "ticks line".
- **Plate Color:** Specify the gauge shape's color.
- **Tick Side:** Specify the tick side location (Left, Right or Both).
- **Number Side:** Specify the number side location (Left, Right or Both).
- **Needle Side:** Specify the gauge's needle side location (Left, Right or Both).
- **Needle Color:** Specify the gauge's needle color.
- **Needle End Color:** Specify the gauge's needle end color.
- **Bar Width:** Set the gauge's bar width. Accepts only numeric value.
- **Ticks Width:** Set the gauge's ticks width. Accepts only numeric value.
- **Needle Type:** Specify the gauge's needle type (Arrow or Line).
- **Needle Width:** Specify the gauge's needle width. Accepts only numeric value.
- **Animation Duration:** Set the gauge's animation duration value when the needle move to the targeted value. Accepts only numeric value.
- **Gauge Animation Rule:** Specify the gauge animation rule (linear, quad, dequad, quint, dequint, cycle, decycle, bounce, debounce, elastic, delastic).
- **Show Value Box:** Switching this property **ON** will display the gauge's value box inside the gauge's control shape.
- **Color Highlights:** Clicking this button will display the gauge's color highlights dialog which consist of the following properties:
  - **From:** Set the highlight "From" value. Accepts only numeric value.
  - **To:** Set the highlight "To" value. Accepts only numeric value.
  - **Color:** Specify the gauge highlight color.
  - **Add New:** You can add multiple highlights by clicking the **Add data** icon  .
- **Style:** The HTML CSS code. The user can edit this code in the editor and change the control style in the designer and art runtime.
- **Operator:** Specify the mathematical operation that you want the gauge control to represent data based on (Sum, Avg, Min, Max and Count).
- **Data Source:** Specify the data source column to be represented in the chart.

- **Show Control:** A dropdown list defines how the designer wants this control to be displayed along report's pages if the report "**Details Page**" generated multiple pages, the designer can choose to have this control **On Every Page, On First Page and On Last Page**. Note that this property will only be visible if the control placed on the **Details Page**, and will operate on generated pages of this report's section only.
- **Hidden:** Specifies if the control will be visible or not when running the report.

Note: Double click on the control to open the **Control Style Properties** panel and design it the way you want. Refer to the **Control Style Properties** part in the **Report Design Ribbon** section for more details.

See also

Reference
addAttribute() function
getAttributeValue() function
modifyAttributeValue() function

9.4 Container Controls

9.4.1 Panel Control

The **Panel** control is used to group controls together and optionally can be designed to have background, style and border around the group.
To group controls together:

1. Drag and drop a **Panel** control onto the report's design workspace.
2. Drag and drop any controls that are to be grouped and place them inside the **Panel** control.
3. Configure the controls as desired.

Note: In design mode, controls grouped within a Panel control can be moved around the report's design workspace collectively.

Control Properties:

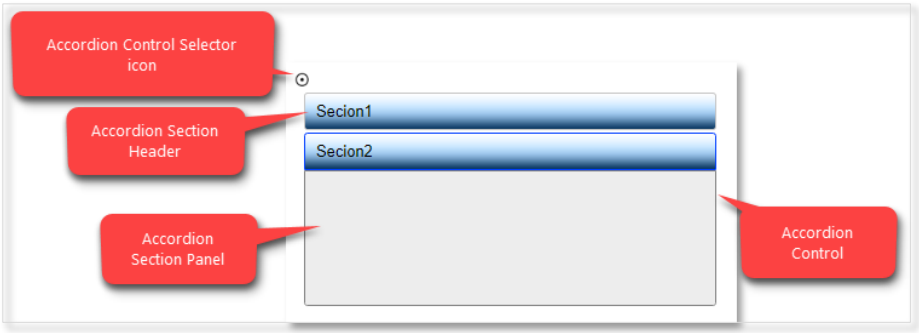
- **Control Type:** The type of the selected control.
- **ID:** The Identity of the control. The ID is a unique value, no duplicate IDs allowed in the report.
- **Tooltip:** Appears on running the report and the user moves the mouse pointer over this control.
- **Layout:** Attribute specifies if the panel control layout will be absolute, relative or Formrelative, if the attribute is absolute the panel will treat the control inside it as absolute position controls (Childs). The designer can move them freely inside the panel, but in some cases i.e. "complex design report" when the designer needs to hide multiple controls and panels depending in multiple complex rules, the relative attribute becomes a necessity in order to arrange these controls inside child panels in the main panel, so when the rules hide or show the panels inside, a main panel they will be shifted up and down freely. The FormRelative option is the same as relative but additionally it has the ability to grow or shrink in height with the report.
- **CSS Classes:** The HTML class attribute makes it possible to define equal styles for elements with the same class name.
- **Style:** The HTML CSS code. The user can edit this code in the editor and change the control style in the designer and art runtime.
- **Show Control:** A dropdown list defines how the designer wants this control to be displayed along report's pages if the report "**Details Page**" generated multiple pages, the designer can choose to have this control **On Every Page, On First Page and On Last Page**. Note that this property will only be visible if the control placed on the **Details Page**, and will operate on generated pages of this report's section only.
- **Disabled:** Specifies if the control will be disabled or not when running the report. This property will affect all contained "child" controls.
- **ReadOnly:** Specifies if the control will be read only or not when running the report.
  This property will affect all contained "child" controls.
- **Hidden:** Specifies if the control will be visible or not when running the report.
  This property will affect all contained "child" controls.

See also

Reference
addAttribute() function
getAttributeValue() function
modifyAttributeValue() function

9.4.2 Accordion Control

An **Accordion** control is used to organize multiple sections into groups; each section consists of a header and a panel; each panel can include a certain type of information or a group of controls to be displayed when the user clicks on the corresponding section header. Opening a section of the **Accordion** control will automatically close the other opened section.



To add an **Accordion** control to the report's design, drag from the toolbox and drop it onto the report's design workspace (canvas).

You can apply the following operations on the **Accordion** control:

- To delete a section from the **Accordion** control, select the section header, then click on delete section ✕ in the control properties.
- To add a section to the **Accordion** control, select on the Accordion Control Selector icon ⊙, then click on Add Section ✚ in the control properties.
- To move a section up and down, select the section and drag it to the desired location then release the mouse.

Accordion Control Properties:
Select the **Accordion** control to set the following properties:

- **Control Type:** The type of the selected control.
- **ID:** The Identity of the control. The ID is a unique value, no duplicate IDs allowed in the report.
- **Tooltip:** Appears on running the report and the user moves the mouse pointer over this control.
- **CSS Classes:** The HTML class attribute makes it possible to define equal styles for elements with the same class name.
- **Style:** The HTML CSS code. The user can edit this code in the editor and change the control style in the designer and art runtime.
- **Add Section:** Add new section in the control.
- **Show Control:** A dropdown list defines how the designer wants this control to be displayed along report's pages if the report "**Details Page**" generated multiple pages, the designer can choose to have this control **On Every Page, On First Page and On Last Page**. Note that this property will only be visible if the control placed on the **Details Page**, and will operate on generated pages of this report's section only.
- **Disabled:** Specifies if the control will be disabled or not when running the report. This property will affect all contained "child" controls.
- **ReadOnly:** Specifies if the control will be read only or not when running the report.
  This property will affect all contained "child" controls.
- **Hidden:** Specifies if the control will be visible or not when running the report.
  This property will affect all contained "child" controls.

Accordion Section Header Properties:
Select the **Accordion Section Header** to set the following properties (to be applied for each Accordion section header):

- **Control Type:** The type of the selected control.
- **ID:** The Identity of the control. The ID is a unique value, no duplicate IDs allowed in the report.
- **Caption:** Set a label text for the selected header section control.
- **Tooltip:** Appears on running the report and the user moves the mouse pointer over this control.
- **CSS Classes:** The HTML class attribute makes it possible to define equal styles for elements with the same class name.
- **Style:** The HTML CSS code. The user can edit this code in the editor and change the control style in the designer and art runtime.
- **Delete Section:** Delete the selected section.

Accordion Section Panel Properties:
Select the **Accordion Section Panel** to set the following properties (to be applied for each Accordion section panel):

- **Control Type:** The type of the selected control.
- **ID:** The Identity of the control. The ID is a unique value, no duplicate IDs allowed in the report.
- **Layout:** Please refer to the Layout property in the **Panel** control.
- **CSS Classes:** The HTML class attribute makes it possible to define equal styles for elements with the same class name.
- **Style:** The HTML CSS code. The user can edit this code in the editor and change the control style in the designer and art runtime.
- **Disabled:** Specifies if the control will be disabled or not when running the report. This property will affect all contained "child" controls.
- **ReadOnly:** Specifies if the control will be read only or not when running the report.
  This property will affect all contained "child" controls.
- **Hidden:** Specifies if the control will be visible or not when running the report.
  This property will affect all contained "child" controls.

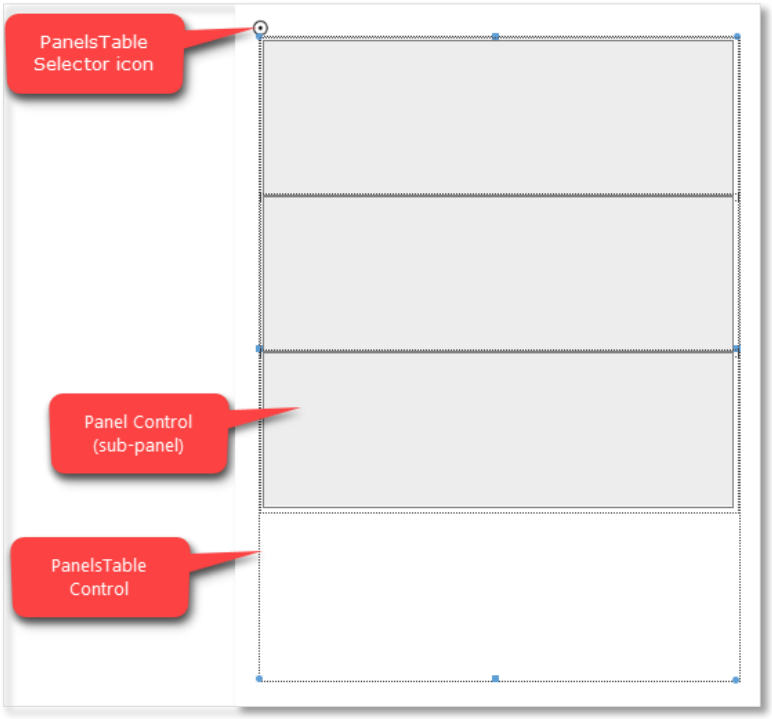
See also

Reference
addAttribute() function
getAttributeValue() function
modifyAttributeValue() function

9.4.3 PanelsTable Control

The **PanelsTable** control is used to control the behavior of different groups of controls inside its panels. The main purpose is to shift controls vertically (up and down) based on their display (showing /hiding).

When a panel inside this control is being hidden, all underneath controls will be shifted up automatically in order to close the gap (spaces) resulted by hiding this control/s and vise versa.



To add a **PanelsTable** control to the form, drag and drop the control onto the report's design workspace (canvas).

You can apply the following operations on the **PanelsTable** control:

- To delete a Panel control from the **PanelsTable** control, select the sub-panel, then press on the Delete key on the keyboard.
- To add a Panel to the **PanelsTable** control, select on the PanelsTable Selector icon ⊙, then click on Add Panel + in the control properties.
- To move a Panel up and down, select the Panel and drag it to the desired location then release the mouse.

Control Properties:
Select the **PanelsTable** control to set the following properties:

- **Control Type:** The type of the selected control.
- **ID:** The Identity of the control. The ID is a unique value, no duplicate IDs allowed in the report.
- **Tooltip:** Appears on running the report and the user moves the mouse pointer over this control.
- **Auto Report Height:** Switch this property ON in order to control the height of the form automatically when shifting the control up and down. In other words, the height of the form will be adjusted by the height of this control.
- **CSS Classes:** The HTML class attribute makes it possible to define equal styles for elements with the same class name.

- **Style:** The HTML CSS code. The user can edit this code in the editor and change the control style in the designer and art runtime.
- **Add Panel:** Add new sub-panel in the control.
- **Show Control:** A dropdown list defines how the designer wants this control to be displayed along report's pages if the report "**Details Page**" generated multiple pages, the designer can choose to have this control **On Every Page, On First Page and On Last Page**. Note that this property will only be visible if the control placed on the **Details Page**, and will operate on generated pages of this report's section only.
- **Disabled:** Specifies if the control will be disabled or not when running the report.
- **ReadOnly:** Specifies if the control will be read only or not when running the report.
- **Hidden:** Specifies if the control will be visible or not when running the report.

Panel Properties:
Select the **Panel** control to set the following properties (to be applied for each panels in the **PanelsTable** control):

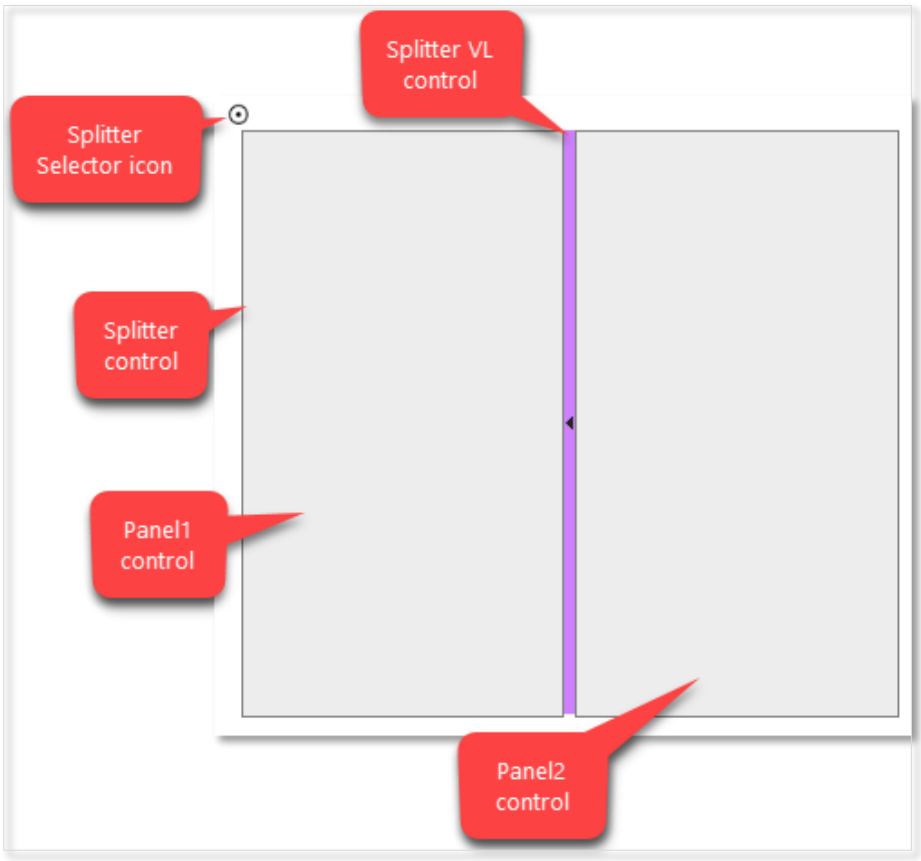
- **Control Type:** The type of the selected control.
- **ID:** The Identity of the control. The ID is a unique value, no duplicate IDs allowed in the report.
- **Tooltip:** Appears on running the report and the user moves the mouse pointer over this control.
- **Layout:** Please refer to the Layout property in the **Panel** control.
- **CSS Classes:** The HTML class attribute makes it possible to define equal styles for elements with the same class name.
- **Style:** The HTML CSS code. The user can edit this code in the editor and change the control style in the designer and art runtime.
- **Disabled:** Specifies if the control will be disabled or not when running the report. This property will affect all contained "child" controls.
- **ReadOnly:** Specifies if the control will be read only or not when running the report.
  This property will affect all contained "child" controls.
- **Hidden:** Specifies if the control will be visible or not when running the report.
  This property will affect all contained "child" controls.

See also

Reference
addAttribute() function
getAttributeValue() function
modifyAttributeValue() function

9.4.4 Splitter Control

The **Splitter** control is a control that represents two panels split by a vertical line (splitter), which allows showing/hiding the left panel when a user clicks on the splitter at runtime. You can place any control on the Splitter's control panels.



The designer can change the style of both panels of the **Splitter** control (i.e. resize the panel width) by double clicking the panel or by selecting the panel and press on **Control Style** in the control ribbon. The same way for changing the style of splitter.

Control Properties:
Select the **Splitter** control to set the following properties:

- **Control Type:** The type of the selected control.
- **ID:** The Identity of the control. The ID is a unique value, no duplicate IDs allowed in the report.
- **Tooltip:** Appears on running the report and the user moves the mouse pointer over this control.
- **CSS Classes:** The HTML class attribute makes it possible to define equal styles for elements with the same class name.
- **Style:** The HTML CSS code. The user can edit this code in the editor and change the control style in the designer and art runtime.
- **Show Control:** A dropdown list defines how the designer wants this control to be displayed along report's pages if the report "**Details Page**" generated multiple pages, the designer can choose to have this control **On Every Page, On First Page and On Last Page**. Note that this property will only be visible if the control placed on the **Details Page**, and will operate on generated pages of this report's section only.
- **Hidden:** Specifies if the control will be visible or not when running the report. This property will affect all contained "child" controls.

Panel Properties:
Select the **Panel** control inside the **Splitter** control to set the following properties (to be applied for both panels in the **Splitter** control):

- **Control Type:** The type of the selected control.
- **ID:** The Identity of the control. The ID is a unique value, no duplicate IDs allowed in the report.
- **Tooltip:** Appears on running the report and the user moves the mouse pointer over this control.
- **Layout:** Please refer to the Layout property in the **Panel** control.
- **CSS Classes:** The HTML class attribute makes it possible to define equal styles for elements with the same class name.
- **Style:** The HTML CSS code. The user can edit this code in the editor and change the control style in the designer and art runtime.
- **Disabled:** Specifies if the control will be disabled or not when running the report. This property will affect all contained "child" controls.
- **ReadOnly:** Specifies if the control will be read only or not when running the report.
  This property will affect all contained "child" controls.
- **Hidden:** Specifies if the control will be visible or not when running the report.
  This property will affect all contained "child" controls.

Splitter VL Properties:
Select the **Splitter VL** control to set the following properties:

- **Control Type:** The type of the selected control.
- **ID:** The Identity of the control. The ID is a unique value, no duplicate IDs allowed in the report.
- **Tooltip:** Appears on running the report and the user moves the mouse pointer over this control.
- **CSS Classes:** The HTML class attribute makes it possible to define equal styles for elements with the same class name.
- **Style:** The HTML CSS code. The user can edit this code in the editor and change the control style in the designer and art runtime.

See also

Reference
addAttribute() function
getAttributeValue() function
modifyAttributeValue() function

9.5 Advanced Controls

9.5.1 Electronic Signature Control

Note: This topic applies to SPARK Reports Builder Enterprise Edition only.

When typically signing a report, you would first need to print the form out, sign it in pen, and then find a way (scanning or faxing) to deliver it where it needs to go. Instead of wasting all that time and paper, Electronic Signature cut out all the steps in between and allows your users to directly and securely sign your report as they are filling it out online.

Control Properties:

- **Control Type:** The type of the selected control.
- **ID:** The Identity of the control. The ID is a unique value, no duplicate IDs allowed in the report.
- **Tooltip:** Appears on running the report and the user moves the mouse pointer over this control.
- **CSS Classes:** The HTML class attribute makes it possible to define equal styles for elements with the same class name.
- **Style:** The HTML CSS code. The user can edit this code in the editor and change the control style in the designer and art runtime.
- **Required:** Specifies if the control will be required to be filled or not when the form loads. This property will not be active at runtime when both **Required** and **Hidden** properties are set to ON.
- **Show Control:** A dropdown list defines how the designer wants this control to be displayed along report's pages if the report "**Details Page**" generated multiple pages, the designer can choose to have this control **On Every Page, On First Page and On Last Page**. Note that this property will only be visible if the control placed on the **Details Page**, and will operate on generated pages of this report's section only.
- **Disabled:** Specifies if the control will be disabled or not when running the report.
- **ReadOnly:** Specifies if the control will be read only or not when running the report.
- **Hidden:** Specifies if the control will be visible or not when running the report.

See also

Reference
addAttribute() function
getAttributeValue() function
modifyAttributeValue() function

9.5.2 Barcode Control

Note: This topic applies to SPARK Reports Builder Enterprise Edition only.

This control quickly and easily add industry-standard barcode to the report been designed through **SPARK Reports Builder**. It is designed to be easy to use and light on resources. It renders Scalable Vector Graphics (SVG) directly into the report's HTML page by specifying encoding barcode standard and text to be encoded. The code can be generated from the default value property or even can be generated randomly by selecting the random option from within its settings.

Control Properties:

- **Control Type:** The type of the selected control.

- **ID:** The Identity of the control. The ID is a unique value, no duplicate IDs allowed in the report.
- **Default Value:** Set a default value for the Barcode. The value will only be used if no value has been specified in the control rules or the random onLoad property is enabled**.**
- **Barcode Type:** Select the Barcode type from the (encoding standards). The shape and the value of the barcode will vary depending on the type selected, also you need to be aware of the barcode values that fits with each type or the barcode will show "Invalid Barcode" message. The random onLoad option will be disabled on certain types as these types have to include a special coding format.
- **Generate onLoad:** By switching it **ON**, the report will generate a random code for the selected barcode control, this value will be automatically generated on the report's load; this option is available for certain types of barcode only.
- **Number of Digits:** This property will be visible when switching the **Generate onLoad** property **ON**, to specify the number of digits that random function will automatically generate.
- **Show Code Value:** Selecting this option will display the barcode value at the bottom of the barcode image.
- **Tooltip:** Appears on running the report and the user moves the mouse pointer over this control.
- **CSS Classes:** The HTML class attribute makes it possible to define equal styles for elements with the same class name.
- **Style:** The HTML CSS code. The user can edit this code in the editor and change the control style in the designer and art runtime.
- **Show Control:** A dropdown list defines how the designer wants this control to be displayed along report's pages if the report "**Details Page**" generated multiple pages, the designer can choose to have this control **On Every Page, On First Page and On Last Page**. Note that this property will only be visible if the control placed on the **Details Page**, and will operate on generated pages of this report's section only.
- **Hidden:** Specifies if the control will be visible or not when running the report.

See also

Reference
addAttribute() function
getAttributeValue() function
modifyAttributeValue() function

9.5.3 QR Control

Note: This topic applies to SPARK Reports Builder Enterprise Edition only.

The QR control adds a QR symbol to the report been designed through **SPARK Reports Builder**. The QR code can be anything from plain text, email address or hyperlink, the code value can be set directly to the control in a static way or dynamically using actions rules.

Control Properties:

- **Control Type:** The type of the selected control.
- **ID:** The Identity of the control. The ID is a unique value, no duplicate IDs allowed in the report.
- **Default Value:** Set a default value for the QR. The value will only be used if no value has been assigned to it in the control actions rules.
- **Tooltip:** Appears on running the report and the user moves the mouse pointer over this control.

- **CSS Classes:** The HTML class attribute makes it possible to define equal styles for elements with the same class name.
- **Style:** The HTML CSS code. The user can edit this code in the editor and change the control style in the designer and art runtime.
- **Show Control:** A dropdown list defines how the designer wants this control to be displayed along report's pages if the report "**Details Page**" generated multiple pages, the designer can choose to have this control **On Every Page, On First Page and On Last Page**. Note that this property will only be visible if the control placed on the **Details Page**, and will operate on generated pages of this report's section only.
- **Hidden:** Specifies if the control will be visible or not when running the report.

See also

Reference
addAttribute() function
getAttributeValue() function
modifyAttributeValue() function
getValue() function

10 Ad-Hoc Query Form Controls

SPARK Report Builder provides users with many various controls to design **Ad-Hoc Query Forms** to provide flexible and scalable report for their users and decisions makers. An ad-hoc query is created to obtain information as the need arises. Contrast with a query that is predefined and routinely processed. **Ad-Hoc Query Form** allows users to control the report's query by passing dynamic parameters to that query which will affect execution returned results. The designer will be able to control how the forms will be interacted with users, for example hide sections of controls based on specific group of users, or provide different parameters layout based on security measures. There are no limits to what users can do using these forms, which empowers them to utilize stored data in a very useful and effective ways.

Note: To submit/execute the form and pass its parameters to the report's query, the user needs to click on the "**Run Report**" button at the top ribbon of the report. Refer to **Running a Report** section for more details.

10.1 General Controls

10.1.1 Label Control

The **Label** control displays text on Ad-Hoc query form. It is usually a static control but can be a dynamic one too. A label is generally used to identify a nearby text box or another widget.

Control Properties:

- **Control Type:** The type of the selected control.
- **ID:** The Identity of the control. The ID is a unique value and no duplicate IDs allowed in the form.
- **Text:** Set a default value for the control. The value will only be used if no value has been specified in the control rules**.** You can click on this icon (Control Text Editor) to add multi-line rich text.
- **Tooltip:** Appears on the running mode when the user moves the mouse pointer over the control.
- **CSS Classes:** The HTML class attribute makes it possible to define equal styles for elements with the same class name.
- **Style:** The HTML style code in a free style editor that can be modified on the spot for the selected control.
- **Hidden:** Specifies if the control will be visible or not when the form loads.

See also

Reference
clear() function
copyFromTo() function
isEmpty() function
isEqual() function
isGreaterEqualThan() function
isGreaterThan() function
isNotEmpty() function
isNotEqual() function
isNumber() function
isSmallerEqualThan() function
isSmallerThan() function
length() function
math() function
not() function
notContain() function

notEndWith() function
notStartWith() function
getValue() function
setValue() function
max() function
min() function
split() function
startWith() function
substring() function
sum() function
toLower() function
toUpper() function
toUpperFirst() function
Trim() function
addAttribute() function
getAttributeValue() function
modifyAttributeValue() function
show() function

10.1.2 HTML Text Control

The **HTML Text** control accepts html and CSS scripts as an input and renders these scripts directly in the form. This allows form designers to provide an advanced data presentation for the users from within the form without the need to inject these scripts into the form's source or use formatting rules to do that.

Control Properties:

- **Control Type:** The type of the selected control.
- **ID:** The Identity of the control. The ID is a unique value and no duplicate IDs allowed in the form.
- **Text:** The text/script of the HTML/CSS that the control will render. You can click on this icon  (Rich Text Editor) to add rich HTML contents using the rich HTML Editor features.
- **Tooltip:** Appears on the running mode when the user moves the mouse pointer over the control.
- **CSS Classes:** The HTML class attribute makes it possible to define equal styles for elements with the same class name.
- **Style:** The HTML CSS code. The user can edit this code in the editor and change the control style in the designer and art runtime.
- **Hidden:** Specifies if the control will be visible or not when the form loads.

See also

Reference
clear() function
copyFromTo() function
isEmpty() function
isEqual() function
isGreaterEqualThan() function
isGreaterThan() function
isNotEmpty() function
isNotEqual() function
isNumber() function
isSmallerEqualThan() function
isSmallerThan() function
length() function
math() function
not() function
notContain() function
notEndWith() function
notStartWith() function

getValue() function
setValue() function
max() function
min() function
split() function
startWith() function
substring() function
sum() function
toLower() function
toUpper() function
toUpperFirst() function
Trim() function
addAttribute() function
getAttributeValue() function
modifyAttributeValue() function
show() function
appendRichTextEditor() function

10.1.3 Horizontal Line control

The **Horizontal Line** control defines a graphical horizontal line in the form's design. For example, it can be used to separate contents in the form.

Control Properties:

- **Control Type:** The type of the selected control.
- **ID:** The Identity of the control. The ID is a unique value and no duplicate IDs allowed in the form.
- **Tooltip:** Appears on the running mode when the user moves the mouse pointer over the control.
- **CSS Classes:** The HTML class attribute makes it possible to define equal styles for elements with the same class name.
- **Style:** The HTML CSS code. The user can edit this code in the editor and change the control style in the designer and art runtime.
- **Hidden:** Specifies if the control will be visible or not when the form loads.

See also

Reference
addAttribute() function
getAttributeValue() function
modifyAttributeValue() function

10.1.4 Vertical Line control

The **Vertical Line** control defines a graphical vertical line in the form design. For example, it can be used to separate contents in the form.

Control Properties:

- **Control Type:** The type of the selected control.
- **ID:** The Identity of the control. The ID is a unique value and no duplicate IDs allowed in the form.
- **Tooltip:** Appears on the running mode when the user moves the mouse pointer over the control.
- **CSS Classes:** The HTML class attribute makes it possible to define equal styles for elements with the same class name.
- **Style:** The HTML CSS code. The user can edit this code in the editor and change the control style in the designer and art runtime.
- **Height type:** Specifies if the control height will be fixed or auto. Auto height will be equal to parent height where parent could be a panel or any Container Controls.

- **Hidden:** Specifies if the control will be visible or not when the form loads.

See also

Reference
addAttribute() function
getAttributeValue() function
modifyAttributeValue() function

10.1.5 Hyperlink Control

The **Hyperlink** control can be used to enter a hyperlink URL and display text.

Control Properties:

- **Control Type:** The type of the selected control.
- **ID:** The Identity of the control. The ID is a unique value and no duplicate IDs allowed in the form.
- **Tab Index:** Attribute specifies the tab order of an element (when the "tab" button is used for navigating).
- **Default value:** Set a default display text for the hyperlink. The value will only be used if no value has been specified in the control rules**.**
- **Tooltip:** Appears on the running mode when the user moves the mouse pointer over the control.
- **Url:** Set the hyperlink URL value for the control. If a # is specified (the default), the current opened URL will be opened in a new tab.
- **Target:** An attribute that specifies where to open the linked item. This property can be one of the following options:
  - _blank: Opens the link in a new window or tab.
  - _parent: Opens the link in the parent frame.
  - _self: Opens the link in the same frame.
  - _top: Opens the link in the full body of the window.
  - _search: Opens the link in the browser's Search pane.
- **Edit HyperLink:** Specify if the user will be able to edit this control at runtime, if this property is checked, then an icon will display next to the control, which will show a dialog of the link text and the URL values to be edited by the user.
- **CSS Classes:** The HTML class attribute makes it possible to define equal styles for elements with the same class name.
- **Style:** The HTML CSS code. The user can edit this code in the editor and change the control style in the designer and art runtime.
- **Hidden:** Specifies if the control will be visible or not when the form loads.

See also

Reference
clear() function
setHyperLink() function
getValue() function
addAttribute() function
getAttributeValue() function
modifyAttributeValue() function

10.1.6 Page Viewer Control

The **Page Viewer** control can be used to display an external page contents in the form. The page viewer control works as an iframe and allows a view of a page or document to be included within the control.

Control Properties:

- **Control Type:** The type of the selected control.
- **ID:** The Identity of the control. The ID is a unique value and no duplicate IDs allowed in the form.
- **Default value:** Sets the initial page URL value for the control. The value will only be used if no value has been specified in the control rules.
- **Tab Index:** Attribute specifies the tab order of an element (when the "tab" button is used for navigating).
- **Tooltip:** Appears on the running mode when the user moves the mouse pointer over the control.
- **CSS Classes:** The HTML class attribute makes it possible to define equal styles for elements with the same class name.
- **Style:** The HTML CSS code. The user can edit this code in the editor and change the control style in the designer and art runtime.
- **Hidden:** Specifies if the control will be visible or not when the form loads.

See also

Reference
clear() function
setPageViewer() function
getValue() function
addAttribute() function
getAttributeValue() function
modifyAttributeValue() function

10.1.7 Image Control

The **Image** control is used to display an image on a form.

Control Properties:

- **Control Type:** The type of the selected control.
- **ID:** The Identity of the control. The ID is a unique value and no duplicate IDs allowed in the form.
- **Tooltip:** Appears on the running mode when the user moves the mouse pointer over the control.
- **Image URL:** Set the URL of the image.
- **CSS Classes:** The HTML class attribute makes it possible to define equal styles for elements with the same class name.
- **Style:** The HTML CSS code. The user can edit this code in the editor and change the control style in the designer and art runtime.
- **Hidden:** Specifies if the control will be visible or not when the form loads.

See also

Reference
clear() function
setImage() function
getValue() function
addAttribute() function
getAttributeValue() function
modifyAttributeValue() function

10.2 Input Controls

10.2.1 Button Control

The Button control can be used to initiate an action, such as executing a custom JavaScript code.

Control Properties:

- **Control Type:** The type of the selected control.
- **ID:** The Identity of the control. The ID is a unique value and no duplicate IDs allowed in the form.
- **Caption:** Set a display text for the button control, you can click on this icon (Control Text Editor) to add multi-line rich text.
- **Tab Index:** Attribute specifies the tab order of an element (when the "tab" button is used for navigating).
- **Tooltip:** Appears on the running mode when the user moves the mouse pointer over the control.
- **CSS Classes:** The HTML class attribute makes it possible to define equal styles for elements with the same class name.
- **Style:** The HTML CSS code. The user can edit this code in the editor and change the control style in the designer and art runtime.
- **Validate OnClick:** Make form validation for not hidden controls. The designer can let the button click action validates all the form's controls and alert the user of any missing data or validation errors. For example.
- **Disabled:** Specifies if the control will be disabled or not when the form loads.
- **ReadOnly:** Specifies if the control will be read only or not when the form loads.
- **Hidden:** Specifies if the control will be visible or not when the form loads.

See also

Reference
clear() function
setFocus() function
setValue() function
addAttribute() function
getAttributeValue() function
modifyAttributeValue() function

10.2.2 Textbox Control

The **Textbox** control allows users to enter plain text on a form.

Control Properties:

- **Control Type:** The type of the selected control.
- **ID:** The Identity of the control. The ID is a unique value and no duplicate IDs allowed in the form.
- **Default value:** Set a default value for the control. The value will only be used if no value has been specified in the control rules. You can click on this icon (Control Text Editor) to add multi-line rich text.
- **Tab Index:** Attribute specifies the tab order of an element (when the "tab" button is used for navigating).
- **Placeholder Text:** A short hint that describes the expected value of an input control.
- **Tooltip:** Appears on the running mode when the user moves the mouse pointer over the control.

- **CSS Classes:** The HTML class attribute makes it possible to define equal styles for elements with the same class name.
- **Style:** The HTML CSS code. The user can edit this code in the editor and change the control style in the designer and art runtime.
- **Number Only:** Attribute specifies if the textbox control must handle numbers only, if this attribute is true the control will not allow typing text inside it, and will map only with SharePoint field of type "Number".
- **Percentage (%):** An attribute which specifies if the control will be shown as a percentage value or not in the form (for example, 50%). This property is visible only when the "**Number Only"** property is switched ON.
- **Decimal Places:** Optional. The number of digits after the decimal point. Default is zero (no digits after the decimal point). This property is visible only when the **Number Only** property is switched ON.
- **Group Separator:** Specifies if the control will be show group separator or not (use 1000 Separator). This property is visible only when the **Number Only** property is switched ON.
- **Required:** Specifies if the control will be required to be filled or not when the form loads. This property will not be active at runtime when both **Required** and **Hidden** properties are set to ON.
- **Required Border:** This property is visible only when the **Required** property is switched ON. This property allows the designer to choose if the red border would appear on the "required" control or not, if not (Required Border is switched OFF) then the designer can keep the default border or change its style as desired.
- **Disabled:** Specifies if the control will be disabled or not when the form loads.
- **ReadOnly:** Specifies if the control will be read only or not when the form loads.
- **Hidden:** Specifies if the control will be visible or not when the form loads.

See also

Reference
clear() function
copyFromTo() function
isEmpty() function
isEqual() function
isGreaterEqualThan() function
isGreaterThan() function
isNotEmpty() function
isNotEqual() function
isNumber() function
isSmallerEqualThan() function
isSmallerThan() function
length() function
math() function
not() function
notContain() function
notEndWith() function
notStartWith() function
getValue() function
setValue() function
max() function
min() function
setFocus() function
split() function
startWith() function
substring() function
sum() function
toLower() function
toUpper() function
toUpperFirst() function
Trim() function
addAttribute() function

getAttributeValue() function
modifyAttributeValue() function
show() function

10.2.3 TextArea Control

The **TextArea** control allows users to enter multiline plain text on a form.

Control Properties:

- **Control Type:** The type of the selected control.
- **ID:** The Identity of the control. The ID is a unique value and no duplicate IDs allowed in the form.
- **Default value:** Set a default value for the control. The value will only be used if no value has been specified in the control rules. You can click on this icon (Control Text Editor) to add multi-line rich text.
- **Tab Index:** Attribute specifies the tab order of an element (when the "tab" button is used for navigating).
- **Placeholder Text:** A short hint that describes the expected value of an input control.
- **Tooltip:** Appears on the running mode when the user moves the mouse pointer over the control.
- **Fixed Height:** Specifies if the control will be fixed height or not. The designer can make the control's height fixed and not expandable and makes the vertical scrollbar appears when the user enters much data "text" in the control.
- **CSS Classes:** The HTML class attribute makes it possible to define equal styles for elements with the same class name.
- **Style:** The HTML CSS code. The user can edit this code in the editor and change the control style in the designer and art runtime.
- **Required:** Specifies if the control will be required to be filled or not when the form loads. This property will not be active at runtime when both **Required** and **Hidden** properties are set to ON.
- **Required Border:** This property is visible only when the **Required** property is switched ON. This property allows the designer to choose if the red border would appear on the "required" control or not. If not (Required Border is switched OFF), the designer can keep the default border or change it as desired.
- **Disabled:** Specifies if the control will be disabled or not when the form loads.
- **ReadOnly:** Specifies if the control will be read only or not when the form loads.
- **Hidden:** Specifies if the control will be visible or not when the form loads.

See also

Reference
clear() function
copyFromTo() function
isEmpty() function
isEqual() function
isGreaterEqualThan() function
isGreaterThan() function
isNotEmpty() function
isNotEqual() function
isNumber() function
isSmallerEqualThan() function
isSmallerThan() function
length() function
math() function
not() function
notContain() function
notEndWith() function

notStartWith() function
getValue() function
setValue() function
max() function
min() function
setFocus() function
split() function
startWith() function
substring() function
sum() function
toLower() function
toUpper() function
toUpperFirst() function
trim() function
addAttribute() function
getAttributeValue() function
modifyAttributeValue() function
show() function

10.2.4 Rich Text Editor Control

The **Rich Text Editor** control allows users to display formatted text, pictures, hyperlinks and tables on a form.

Control Properties:

- **Control Type:** The type of the selected control.
- **ID:** The Identity of the control. The ID is a unique value and no duplicate IDs allowed in the form.
- **Default value:** Set a default value for the control. The value will only be used if no value has been specified in the control rules. You can click on this icon (Control Text Editor) to add multi-line rich text.
- **Tab Index:** Attribute specifies the tab order of an element (when the "tab" button is used for navigating).
- **Tooltip:** Appears on the running mode when the user moves the mouse pointer over the control.
- **Fixed Height:** Specifies if the control will be fixed height or not.
- **CSS Classes:** The HTML class attribute makes it possible to define equal styles for elements with the same class name.
- **Style:** The HTML CSS code. The user can edit this code in the editor and change the control style in the designer and art runtime.
- **Required:** Specifies if the control will be required to be filled or not when the form loads. This property will not be active at runtime when both **Required** and **Hidden** properties are set to ON.
- **Required Border:** This property is visible only when the **Required** property is switched ON. This property allows the designer to choose if the red border would appear on the "required" control or not. If not (Required Border is switched OFF), the designer can keep the default border or change it as desired.
- **Disabled:** Specifies if the control will be disabled or not when the form loads.
- **ReadOnly:** Specifies if the control will be read only or not when the form loads.
- **Hidden:** Specifies if the control will be visible or not when the form loads.

See also

Reference
clear() function
copyFromTo() function
isEmpty() function
isEqual() function
isGreaterEqualThan() function
isGreaterThan() function
isNotEmpty() function

isNotEqual() function
isNumber() function
isSmallerEqualThan() function
isSmallerThan() function
length() function
math() function
not() function
notContain() function
notEndWith() function
notStartWith() function
getValue() function
setValue() function
max() function
min() function
setFocus() function
split() function
startWith() function
substring() function
sum() function
toLower() function
toUpper() function
toUpperFirst() function
Trim() function
addAttribute() function
getAttributeValue() function
modifyAttributeValue() function
show() function
appendRichTextEditor() function

10.2.5 CheckboxList Control

The **CheckboxList** control can be used to make a single or multiple selection on a form.

Control Properties:

- **Control Type:** The type of the selected control.
- **ID:** The Identity of the control. The ID is a unique value and no duplicate IDs allowed in the form.
- **Tab Index**: Attribute specifies the tab order of an element (when the "tab" button is used for navigating).
- **Tooltip:** Appears on the running mode when the user moves the mouse pointer over the control.
- **Fixed Height:** Specify if the control will be fixed height or not.
- **CSS Classes:** The HTML class attribute makes it possible to define equal styles for elements with the same class name.
- **Style:** The HTML CSS code. The user can edit this code in the editor and change the control style in the designer and art runtime.
- **Alignment:** Specify how the control would show and list its items (vertically or horizontally).
- **Items:** You can specify items that will be shown in the control (Statically), note that you need to click apply button to reflect these items in the control.
- **Reverse Items:** Reverse the listed items sorting.
- **Fill Choice:** Set this property to ON to enable users adding their own choice.
- **Fill Choice Message:** This property is visible if the **Fill Choice** property set to ON. Setting this property to "ON", will let the designer specifies the Fill Choice Message and replace it with the default one. This property accepts html tags.
- **Reflect Inside Control:** This property is visible if the **Fill Choice** property set to ON. Setting this property to ON will let the entered value in the textbox choice to be reflected as a checkbox item when pressing the enter key.

- **Required:** Specifies if the control will be required to be filled or not when the form loads. This property will not be active at runtime when both **Required** and **Hidden** properties are set to ON.
- **Required Border:** This property is visible only when the **Required** property is switched ON. This property allows the designer to choose if the red border would appear on the "required" control or not. If not (Required Border is switched OFF), the designer can keep the default border or change it as desired.
- **Disabled:** Specifies if the control will be disabled or not when the form loads.
- **Hidden:** Specifies if the control will be visible or not when the form loads.

See also

Reference
clear() function
addCheckBoxListItem() function
getCheckBoxListItems() function
removeCheckBoxListItemByIndex() function
removeCheckBoxListItemByName() function
selectAtLeast() function
selectAtMost() function
selectExactly() function
setCheckBoxListItemByIndex() function
setCheckBoxListItemByName() function
setFocus() function
addAttribute() function
getAttributeValue() function
modifyAttributeValue() function

10.2.6 Checkbox Control

The **Checkbox** control permits the user to make a binary choice, i.e. a choice between one of two possible mutually exclusive options. For example, the user may have to answer 'yes' (checked) or 'no' (not checked) on a simple yes/no question.

Control Properties:

- **Control Type:** The type of the selected control.
- **ID:** The Identity of the control. The ID is a unique value and no duplicate IDs allowed in the form.
- **Default value:** Set a default value for the control (checked/ Unchecked). The value will only be used if no value has been specified in the control rules**.**
- **Caption:** Set a label text for the checkbox control.
- **Tab Index:** attribute specifies the tab order of an element (when the "tab" button is used for navigating).
- **Tooltip:** Appears on the running mode when the user moves the mouse pointer over the control.
- **CSS Classes:** The HTML class attribute makes it possible to define equal styles for elements with the same class name.
- **Style:** The HTML CSS code. The user can edit this code in the editor and change the control style in the designer and art runtime.
- **Disabled:** Specifies if the control will be disabled or not when the form loads.
- **Hidden:** Specifies if the control will be visible or not when the form loads.

See also

Reference
clear() function
getCheckBox() function
setCheckBox() function

setFocus() function
addAttribute() function
getAttributeValue() function
modifyAttributeValue() function

10.2.7 DropDownList Control

The **DropdownList** control is a graphical control element, similar to a list box that allows the user to choose one value from a list. When a drop-down list is inactive, it displays a single value. When activated (clicked on) it displays (drops down) a list of values, from which the user may select one.

Control Properties:

- **Control Type:** The type of the selected control.
- **ID:** The Identity of the control. The ID is a unique value and no duplicate IDs allowed in the form.
- **Tab Index:** Attribute specifies the tab order of an element (when the "tab" button is used for navigating).
- **Tooltip:** Appears on the running mode when the user moves the mouse pointer over the control.
- **CSS Classes:** The HTML class attribute makes it possible to define equal styles for elements with the same class name.
- **Style:** The HTML CSS code. The user can edit this code in the editor and change the control style in the designer and art runtime.
- **Items:** You can specify the items that will be shown in the control (Static), you need to click apply button to reflect these items in the control.
- **Fill Choice:** Set this property to ON to enable users adding their own choice.
- **Fill Choice Message:** This property is visible if the **Fill Choice** property set to ON. Setting this property to "ON", will let the designer specifies the Fill Choice Message and replace it with the default one. This property accepts html tags.
- **Reflect Inside Control:** This property is visible if the **Fill Choice** property set to ON. Setting this property to ON will let the entered value in the textbox choice to be reflected as a radio button item when pressing the enter key.
- **Required:** Specifies if the control will be required to be filled or not when the form loads. This property will not be active at runtime when both **Required** and **Hidden** properties are set to ON.
- **Required Border:** This property is visible only when the **Required** property is switched ON. This property allows the designer to choose if the red border would appear on the "required" control or not. If not (Required Border is switched OFF), the designer can keep the default border or change it as desired.
- **Disabled:** Specifies if the control will be disabled or not when the form loads.
- **Hidden:** Specifies if the control will be visible or not when the form loads.

See also

Reference
clear() function
addDropDownItem() function
removeDropDownItem() function
isEmpty() function
isEqual() function
isGreaterEqualThan() function
isGreaterThan() function
isNotEmpty() function
isNotEqual() function
isNumber() function
isSmallerEqualThan() function
isSmallerThan() function
length() function
math() function

not() function
notContain() function
notEndWith() function
notStartWith() function
getValue() function
setValue() function
max() function
min() function
setFocus() function
split() function
startWith() function
substring() function
sum() function
toLower() function
toUpper() function
toUpperFirst() function
Trim() function
addAttribute() function
getAttributeValue() function
modifyAttributeValue() function
show() function

10.2.8 Date Control

The **Date** control is used to enter a date manually or from a calendar display.

Control Properties:

- **Control Type:** The type of the selected control.
- **ID:** The Identity of the control. The ID is a unique value and no duplicate IDs allowed in the form.
- **Default value:** Set a default value for the control. The value will only be used if no value has been specified in the control rules**.**
- **Tab Index:** Attribute specifies the tab order of an element (when the "tab" button is used for navigating).
- **Tooltip:** Appears on the running mode when the user moves the mouse pointer over the control.
- **CSS Classes:** The HTML class attribute makes it possible to define equal styles for elements with the same class name.
- **Style:** The HTML CSS code. The user can edit this code in the editor and change the control style in the designer and art runtime.
- **Format:** Set date format for the date control, the default format is mm/dd/yy.
- **Min:** Specify the minimum valid date value. The entered date value will be validated against a specified minimum value.
- **Max:** Specify the maximum valid date value. The entered date value will be validated against a specified maximum value.
- **Required:** Specifies if the control will be required to be filled or not when the form loads. This property will not be active at runtime when both **Required** and **Hidden** properties are set to ON.
- **Required Border:** This property is visible only when the **Required** property is switched ON. This property allows the designer to choose if the red border would appear on the "required" control or not. If not (Required Border is switched OFF), the designer can keep the default border or change it as desired.
- **Disabled:** Specifies if the control will be disabled or not when the form loads.
- **ReadOnly:** Specifies if the control will be read only or not when the form loads.
- **Hidden:** Specifies if the control will be visible or not when the form loads.

Examples:

Input Date	Format	Result
10/10/2017	M, dd y	Oct, 10 17

Input Date	Format	Result
10/10/2017	MM, dd yy	Oct, 10 2017
10/10/2017	MM, dd y	October, 10 17
10/10/2017	MM, dd yy	October, 10 2017
10/03/2017	MM, d y	October, 3 17
10/03/2017	mm/dd/yy	10/03/2017
10/03/2017	mm/d/yy	10/3/2017

See also

Reference
clear() function
currentDate() function
calculateDays() function
isEmpty() function
isEqual() function
isGreaterEqualThan() function
isGreaterThan() function
isNotEmpty() function
isNotEqual() function
isNumber() function
isSmallerEqualThan() function
isSmallerThan() function
length() function
not() function
notContain() function
notEndWith() function
notStartWith() function
getValue() function
setValue() function
setFocus() function
split() function
startWith() function
substring() function
toLower() function
toUpper() function
toUpperFirst() function
Trim() function
addAttribute() function
getAttributeValue() function
modifyAttributeValue() function
show() function
calculateDaysExcludedDays() function
weekendBetweenTwoDates function
differenceDateTime() function
extractDateValue() function

10.2.9 Time Control

The **Time** control is used for entering a time as hh:mm.

Control Properties:

- **Control Type:** The type of the selected control.
- **ID:** The Identity of the control. The ID is a unique value and no duplicate IDs allowed in the form.
- **Default value:** Set a default value for the control. The value will only be used if no value has been specified in the control rules**.**
- **Tab Index:** Attribute specifies the tab order of an element (when the "tab" button is used for navigating).
- **Tooltip:** Appears on the running mode when the user moves the mouse pointer over the control.

- **CSS Classes:** The HTML class attribute makes it possible to define equal styles for elements with the same class name.
- **Style:** The HTML CSS code. The user can edit this code in the editor and change the control style in the designer and art runtime.
- **Format:** Set time format for the time control, the default format is H:i.
- **Time Step:** Set time step for the time control, the default value is 60 (means 60 minute). For example, if you specify the time step 60, the list of values in the control in the running mode will be 00:00, 01:00, 02:00, 03:00, 02:00, 03:00 and so on. If you specify the time step 30, the list of values in the control in the running mode will be 00:00, 00:30, 01:00, 01:30, 02:00, 02:30 and so on.
- **Required:** Specifies if the control will be required to be filled or not when the form loads. This property will not be active at runtime when both **Required** and **Hidden** properties are set to ON.
- **Required Border:** This property is visible only when the **Required** property is switched ON. This property allows the designer to choose if the red border would appear on the "required" control or not. If not (Required Border is switched OFF), the designer can keep the default border or change it as desired.
- **Disabled:** Specifies if the control will be disabled or not when the form loads.
- **ReadOnly:** Specifies if the control will be read only or not when the form loads.
- **Hidden:** Specifies if the control will be visible or not when the form loads.

See also

Reference
clear() function
isEmpty() function
isEqual() function
isGreaterEqualThan() function
isGreaterThan() function
isNotEmpty() function
isNotEqual() function
isNumber() function
isSmallerEqualThan() function
isSmallerThan() function
length() function
not() function
notContain() function
notEndWith() function
notStartWith() function
getValue() function
setValue() function
setFocus() function
split() function
startWith() function
substring() function
toLower() function
toUpper() function
toUpperFirst() function
Trim() function
addAttribute() function
getAttributeValue() function
modifyAttributeValue() function
show() function
addDateTime() function

10.2.10 DateTime Control

The **DateTime** control is used to enter a DateTime manually or from a calendar display.

Control Properties

- **Control Type:** The type of the selected control.
- **ID:** The Identity of the control. The ID is a unique value and no duplicate IDs allowed in the form.
- **Default value:** Set a default value for the control. The value will only be used if no value has been specified in the control rules**.**
- **Tab Index:** Attribute specifies the tab order of an element (when the "tab" button is used for navigating).
- **Tooltip:** Appears on the running mode when the user moves the mouse pointer over the control.
- **CSS Classes:** The HTML class attribute makes it possible to define equal styles for elements with the same class name.
- **Style:** The HTML CSS code. The user can edit this code in the editor and change the control style in the designer and art runtime.
- **Format:** Set date format for the date control, the default format is m/d/Y H:i.
- **Min:** Specify the minimum valid date value (only valid date without time). The entered date value will be validated against a specified minimum value.
- **Max:** Specify the maximum valid date value (only valid date without time). The entered date value will be validated against a specified maximum value.
- **Time Step:** Set time step for the datetime control, the default value is 60 (means 60 minute). For example, if you specify the time step 60, the list of values in the control in the running mode will be 00:00, 01:00, 02:00, 03:00, 02:00, 03:00 and so on. If you specify the time step 30, the list of values in the control in the running mode will be 00:00, 00:30, 01:00, 01:30, 02:00, 02:30 and so on.
- **Required:** Specifies if the control will be required to be filled or not when the form loads. This property will not be active at runtime when both **Required** and **Hidden** properties are set to ON.
- **Required Border:** This property is visible only when the **Required** property is switched ON. This property allows the designer to choose if the red border would appear on the "required" control or not. If not (Required Border is switched OFF), the designer can keep the default border or change it as desired.
- **Disabled:** Specifies if the control will be disabled or not when the form loads.
- **ReadOnly:** Specifies if the control will be read only or not when the form loads.
- **Hidden:** Specifies if the control will be visible or not when the form loads.

The following characters are recognized in the Format property of the DateTime control. The format specified in the Format property should be a correct DateTime format (please see examples below).

Format	Description
Day	
F	A full textual representation of a month, such as January or March
M	Numeric representation of a month, with leading zeros
M	A short textual representation of a month, three letters
N	Numeric representation of a month, without leading zeros
Year	
Y	A full numeric representation of a year, 4 digits
Y	A two digit representation of a year
Time	
A	Lowercase Ante meridiem and Post meridiem
A	Uppercase Ante meridiem and Post meridiem
B	Swatch Internet time
G	12-hour format of an hour without leading zeros

Format	Description
G	24-hour format of an hour without leading zeros
H	12-hour format of an hour with leading zeros
H	24-hour format of an hour with leading zeros
i	Minutes with leading zeros
s	Seconds, with leading zeros
u	Microseconds. Will always generate 000000.

Examples:

Input Date	Format	Result
10/10/2017	m/d/Y H:i:s a	10/10/2017 11:52:25 am
10/10/2017	F d, Y	October 10, 2017
10/10/2017	M d, Y	Oct 10, 2017
10/10/2017	M d, y	Oct 10, 17
10/10/2017	M d, y g:i	Oct 10, 17 12:03
10/10/2017	M d, y g:i:s	Oct 10, 17 12:04:23
10/10/2017	m/d/Y H:i:s a	10/10/2017 12:13:09 pm

See also

Reference
clear() function
currentDate() function
setFocus() function
addAttribute() function
getAttributeValue() function
modifyAttributeValue() function
differenceDateTime() function
extractDateValue() function
addDateTime() function
calculateDaysExcludedDays() function
weekendBetweenTwoDates()

10.2.11 Radio Button Control

The **Radio Button** control is used to make a single or multiple selections in the form.

Control Properties:

- **Control Type:** The type of the selected control.
- **ID:** The Identity of the control. The ID is a unique value and no duplicate IDs allowed in the form.
- **Tab Index:** Attribute specifies the tab order of an element (when the "tab" button is used for navigating).
- **Tooltip:** Appears on the running mode when the user moves the mouse pointer over the control.
- **CSS Classes:** The HTML class attribute makes it possible to define equal styles for elements with the same class name.
- **Style:** The HTML CSS code. The user can edit this code in the editor and change the control style in the designer and art runtime.
- **Alignment:** Specifies the vertical/Horizontal alignment of the Radio buttons collection in the form.
- **Items:** You will be able to specify the items that will be shown in the control (Static), you need to click apply button to reflect these items in the control.
- **Fill Choice:** Set this property to ON to enable users adding their own choice.
- **Fill Choice Message:** This property is visible if the **Fill Choice** property set to ON. Setting this property to "ON", will let the designer specifies the Fill Choice Message and replace it with the default one. This property accepts html tags.

- **Reflect Inside Control:** This property is visible if the **Fill Choice** property set to ON. Setting this property to ON will let the entered value in the textbox choice to be reflected as a radio button item when pressing the enter key.
- **Required:** Specifies if the control will be required to be filled or not when the form loads. This property will not be active at runtime when both **Required** and **Hidden** properties are set to ON.
- **Required Border:** This property is visible only when the **Required** property is switched ON. This property allows the designer to choose if the red border would appear on the "required" control or not. If not (Required Border is switched OFF), the designer can keep the default border or change it as desired.
- **Disabled:** Specifies if the control will be disabled or not when the form loads.
- **ReadOnly:** Specifies if the control will be read only or not when the form loads.
- **Hidden:** Specifies if the control will be visible or not when the form loads.

See also

Reference
setRadioButton() function
getRadioButton() function
setFocus() function
addAttribute() function
getAttributeValue() function
modifyAttributeValue() function

10.2.12 Toggle Switch Control

The **Toggle Switch** control allows designers to add a switch [on/off] control in their forms in order to provide their users with a simple, easy to use and awesome input control in order to control several aspects in the form such as hiding/showing other controls or containers, or return selections (true/false) based on the forms criteria. This control makes the form very clear and fast to fill out for users.

Control Properties:

- **Control Type:** The type of the selected control.
- **ID:** The Identity of the control. The ID is a unique value and no duplicate IDs allowed in the form.
- **Default value:** Set a default value for the control (checked/ Unchecked). The value will only be used if no value has been specified in the control rules**.**
- **Tab Index:** Attribute specifies the tab order of an element (when the "tab" button is used for navigating).
- **Tooltip:** Appears on the running mode when the user moves the mouse pointer over the control.
- **CSS Classes:** The HTML class attribute makes it possible to define equal styles for elements with the same class name.
- **Style:** The HTML CSS code. The user can edit this code in the editor and change the control style in the designer and art runtime.
- **Disabled:** Specifies if the control will be disabled or not when the form loads.
- **Hidden:** Specifies if the control will be visible or not when the form loads.

Control Style Properties

- **Left x Top (px):** Set the left and top positions in pixels for the selected control.
- **ON Color:** Set the toggle switch on (ON Switch) Color for the selected control.
- **OFF Color:** Set the toggle switch on (OFF Switch) Color for the selected control.

See also

Reference
setFocus() function
getValue() function

addAttribute() function
getAttributeValue() function
modifyAttributeValue() function

10.2.13 Currency Control

The **Currency** control allows the designer to create a currency formatted object in the form and allow the users to deal with it as currency value in the form.

Control Property:

- **Control Type:** The type of the selected control.
- **ID:** The Identity of the control. The ID is a unique value and no duplicate IDs allowed in the form.
- **Default value:** Set a default value for the control. The value will only be used if no value has been specified in the control rules.
- **Tab Index**: Attribute specifies the tab order of an element (when the "tab" button is used for navigating).
- **Tooltip:** Appears on the running mode when the user moves the mouse pointer over the control.
- **CSS Classes**: The HTML class attribute makes it possible to define equal styles for elements with the same class name.
- **Style**: The HTML CSS code. The user can edit this code in the editor and change the control style in the designer and art runtime.
- **Static Format:** A dropdown list contains the currency format to bind the input control with.
- **Required:** Specifies if the control will be required to be filled or not when the form loads. This property will not be active at runtime when both **Required** and **Hidden** properties are set to ON.
- **Required Border:** This property is visible only when the **Required** property is switched ON. This property allows the designer to choose if the red border would appear on the "required" control or not. If not (Required Border is switched OFF), the designer can keep the default border or change it as desired.
- **Disabled:** Specifies if the control will be disabled or not when the form loads.
- **ReadOnly:** Specifies if the control will be read only or not when the form loads.
- **Hidden:** Specifies if the control will be visible or not when the form loads.

See also

Reference
clear() function
copyFromTo() function
isEmpty() function
isEqual() function
isGreaterEqualThan() function
isGreaterThan() function
isNotEmpty() function
isNotEqual() function
isNumber() function
isSmallerEqualThan() function
isSmallerThan() function
length() function
math() function
not() function
notContain() function
notEndWith() function
notStartWith() function
getValue() function
setValue() function
max() function

min() function
setFocus() function
split() function
startWith() function
substring() function
sum() function
addAttribute() function
getAttributeValue() function
modifyAttributeValue() function
show() function
getCurrencyValue() function

10.2.14 Slider Control

The **Slider** control is a composite control with buttons and html elements that enable a user to set a numerical range or select from a set of values visually without the need to enter any number manually. Using this tool, the form's designer will no longer need to set validation rules on entering a numeric value in the field.

Control Property:

- **Control Type:** The type of the selected control.
- **ID:** The Identity of the control. The ID is a unique value and no duplicate IDs allowed in the form.
- **Default value:** Set a default value for the control. The value will only be used if no value has been specified in the control.
- **Tooltip:** Appears on the running mode when the user moves the mouse pointer over the control.
- **CSS Classes**: The HTML class attribute makes it possible to define equal styles for elements with the same class name.
- **Style**: The HTML CSS code. The user can edit this code in the editor and change the control style in the designer and art runtime.
- **Skin:** A dropdown list contains a different type of styles that ken applied on the control [Big, Flat, Modern, Round, Sharp and Square].
- **Min-Max:** Specifies the range of numeric values that the control will work on, the Min value must be less than the Max value.
- **Show Min/Max Label:** Switch ON to show the Min and Max labels values at both ends of Slider control.
- **Step:** The numeric value of the Slider step, which represents the difference between each value in the slider grid.
- **Prefix:** (Optional) represents the text that will be shown at the left side of the value label.
- **Postfix:** (Optional) represents the text that will be shown at the right side of the value label.
- **Show Grid:** Switch ON to show the Slider grid values.
- **Grid Unit:** The numeric value of how many labels/values will be shown in the Slider grid.
- **Disabled:** Specifies if the control will be disabled or not when the form loads.
- **ReadOnly:** Specifies if the control will be read only or not when the form loads.
- **Hidden:** Specifies if the control will be visible or not when the form loads.

See also

Reference
clear() function
copyFromTo() function
isEmpty() function
isEqual() function
isGreaterEqualThan() function
isGreaterThan() function

isNotEmpty() function
isNotEqual() function
isNumber() function
isSmallerEqualThan() function
isSmallerThan() function
length() function
math() function
not() function
notContain() function
notEndWith() function
notStartWith() function
getValue() function
setValue() function
max() function
min() function
setFocus() function
split() function
startWith() function
substring() function
sum() function
addAttribute() function
getAttributeValue() function
modifyAttributeValue() function
show() function
getCurrencyValue() function

10.2.15 Round Slider Control

The **Round Slider** control is a composite control with buttons and html elements with rounded shape like a radio volume control, that enable a user to set a numerical range or select from a set of values visually without the need to enter any number manually. Using this tool the form's designer will no longer need to set validation rules on entering a numeric value in the field.

Control Property:

- **Control Type:** The type of the selected control.
- **ID:** The Identity of the control. The ID is a unique value and no duplicate IDs allowed in the form.
- **Default value:** Set a default value for the control. The value will only be used if no value has been specified in the control rules.
- **Tooltip:** Appears on the running mode when the user moves the mouse pointer over the control.
- **CSS Classes**: The HTML class attribute makes it possible to define equal styles for elements with the same class name.
- **Style**: The HTML CSS code. The user can edit this code in the editor and change the control style in the designer and art runtime.
- **Min-Max:** Specifies the range of numeric values that the control will work on, the Min value must be less than the Max value.
- **Step:** The numeric value of the Slider step, which represents the difference between each value in the slider grid.
- **Min-Range:** Switch ON to show the Min area that the control has moved by.
- **Rang Color:** This property is visible only when the **Min-Range** property is switched ON. This property specifies the color of the Min-Range area.
- **Handle Shape:** A dropdown list contains a different type of styles that can be applied on the Round Slider handle [Round, Dot and Square].
- **Handle Color:** Specifies the Round Slider handle color.
- **Radius:** The numeric value of the Round Slider radius size.
- **Path Width:** The numeric value of the Round Slider radius width.

- **Path Color:** Specifies Round Slider path color.
- **Start Angle:** The numeric value of the Round Slider start angle.
- **End Angle:** The numeric value of the Round Slider end angle.
- **Disabled:** Specifies if the control will be disabled or not when the form loads.
- **ReadOnly:** Specifies if the control will be read only or not when the form loads.
- **Hidden:** Specifies if the control will be visible or not when the form loads.

See also

Reference
clear() function
copyFromTo() function
isEmpty() function
isEqual() function
isGreaterEqualThan() function
isGreaterThan() function
isNotEmpty() function
isNotEqual() function
isNumber() function
isSmallerEqualThan() function
isSmallerThan() function
length() function
math() function
not() function
notContain() function
notEndWith() function
notStartWith() function
getValue() function
setValue() function
max() function
min() function
setFocus() function
split() function
startWith() function
substring() function
sum() function
addAttribute() function
getAttributeValue() function
modifyAttributeValue() function
show() function
getCurrencyValue() function

10.3 Container Controls

10.3.1 Panel Control

The **Panel** control is used to group controls together and optionally can be designed to have background, style and border around the group.
To group controls together:

4. Drag and drop a **Panel** control onto the form design workspace.
5. Drag and drop any controls that are to be grouped and place them inside the **Panel** control.
6. Configure the controls as desired.

Note: In design mode, controls grouped within a Panel control can be moved around the form design workspace collectively.

Control Properties:

- **Control Type:** The type of the selected control.
- **ID:** The Identity of the control. The ID is a unique value and no duplicate IDs allowed in the form.
- **Tooltip:** Appears on the running mode when the user moves the mouse pointer over the control.
- **Layout:** Attribute specifies if the panel control layout will be absolute, relative or "Formrelative", if the attribute is absolute the panel will treat the control inside it as absolute position controls (Childs). The designer can move them freely inside the panel, but in some cases i.e. "complex design forms" when the designer needs to hide multiple controls and panels depending in multiple complex rules, the relative attribute becomes a necessity in order to arrange these controls inside child panels in the main panel, so when the rules hide or show the panels inside, a main panel they will be shifted up and down freely. The FormRelative option is the same as relative but additionally it has the ability to grow or shrink in height with the form.
- **CSS Classes:** The HTML class attribute makes it possible to define equal styles for elements with the same class name.
- **Style:** The HTML CSS code. The user can edit this code in the editor and change the control style in the designer and art runtime.
- **Disabled:** Specifies if the control will be disabled or not when the form loads.
- **ReadOnly:** Specifies if the control will be read only or not when the form loads.
- **Hidden:** Specifies if the control will be visible or not when the form loads.

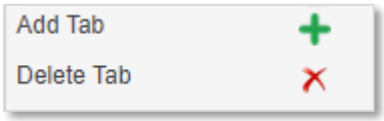
See also

Reference
addAttribute() function
getAttributeValue() function
modifyAttributeValue() function

10.3.2 Tab Control

The **Tab** control is a web control that can define multiple panels in the same location; each tab can consist of a certain type of information or a group of controls that the application displays when the user selects the corresponding tab.
To group controls together:

1. Drag and drop a **Tab** control onto the Forms Designer.
2. Add, edit or remove tabs by using the control property panel where you can find the following buttons to do that.



3. Drag and drop controls, these controls can to be grouped and placed inside each **Tab**.
4. Configure the controls as desired.
5. You can move and sort the internal tabs inside the Tab control by dragging & dropping them.

Note: In the design mode, controls grouped within a Tab control can be moved around the form design workspace collectively.

Control Properties:

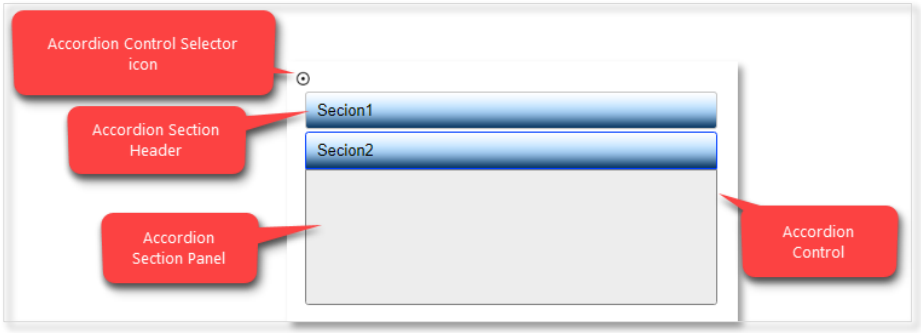
- **Control Type:** The type of the selected control.
- **ID:** The Identity of the control. The ID is a unique value and no duplicate IDs allowed in the form.
- **Caption:** Set a label text for the tab control selected.
- **Tooltip:** Appears on the running mode when the user moves the mouse pointer over the control.
- **Add Tab:** Add new tab in the control.
- **Delete Tab:** Remove the selected tab from the control design.
- **Tab Direction:** Specifies the direction of tabs in the control (Left-Right or Right-Left).
- **Active Tab Color:** Specifies the color of the active tab element, the value should be hexadecimal. You can click on the color icon and select the desired color from the color selector dialog.
- **CSS Classes:** The HTML class attribute makes it possible to define equal styles for elements with the same class name.
- **Style:** The HTML CSS code. The user can edit this code in the editor and change the control style in the designer and art runtime.
- **Disabled:** Specifies if the control will be disabled or not when the form loads.
- **ReadOnly:** Specifies if the control will be read only or not when the form loads.
- **Hidden:** Specifies if the control will be visible or not when the form loads.

See also

Reference
addAttribute() function
getAttributeValue() function
modifyAttributeValue() function
disableTab() function
hideTab() function

10.3.3 Accordion Control

An **Accordion** control is an alternative to Tab control, used to organize multiple sections into groups; each section consists of a header and a panel; each panel can include a certain type of information or a group of controls to be displayed when the user clicks on the corresponding section header. Opening a section of the **Accordion** control will automatically close the other opened section.



To add an **Accordion** control to the form, drag and drop the control onto the form canvas.

You can apply the following operations on the **Accordion** control:

- To delete a section from the **Accordion** control, select the section header, then click on delete section in the control properties.
- To add a section to the **Accordion** control, select on the Accordion Control Selector icon, then click on Add Section in the control properties.
- To move a section up and down, select the section and drag it to the desired location then release the mouse.

Accordion Control Properties:

Select the **Accordion** control to set the following properties:

- **Control Type:** The type of the selected control.
- **ID:** The Identity of the control. The ID is a unique value and no duplicate IDs allowed in the form.
- **Tooltip:** Appears on the running mode when the user moves the mouse pointer over the control.
- **CSS Classes:** The HTML class attribute makes it possible to define equal styles for elements with the same class name.
- **Style:** The HTML CSS code. The user can edit this code in the editor and change the control style in the designer and art runtime.
- **Add Section:** Add new section in the control.
- **Disabled:** Specifies if the control will be disabled or not when the form loads.
- **ReadOnly:** Specifies if the control will be read only or not when the form loads.
- **Hidden:** Specifies if the control will be visible or not when the form loads.

Accordion Section Header Properties:

Select the **Accordion Section Header** to set the following properties (to be applied for each Accordion section header):

- **Control Type:** The type of the selected control.
- **ID:** The Identity of the control. The ID is a unique value and no duplicate IDs allowed in the form.
- **Caption:** Set a label text for the selected header section control.
- **Tooltip:** Appears on the running mode when the user moves the mouse pointer over the control.
- **CSS Classes:** The HTML class attribute makes it possible to define equal styles for elements with the same class name.
- **Style:** The HTML CSS code. The user can edit this code in the editor and change the control style in the designer and art runtime.
- **Delete Section:** Delete the selected section.

Accordion Section Panel Properties:

Select the **Accordion Section Panel** to set the following properties (to be applied for each Accordion section panel):

- **Control Type:** The type of the selected control.
- **ID:** The Identity of the control. The ID is a unique value and no duplicate IDs allowed in the form.
- **Layout:** Please refer to the Layout property in the **Panel** control.
- **CSS Classes:** The HTML class attribute makes it possible to define equal styles for elements with the same class name.
- **Style:** The HTML CSS code. The user can edit this code in the editor and change the control style in the designer and art runtime.
- **Disabled:** Specifies if the control will be disabled or not when the form loads.
- **ReadOnly:** Specifies if the control will be read only or not when the form loads.
- **Hidden:** Specifies if the control will be visible or not when the form loads. Set this property to ON, will hide the panel section and its related header.

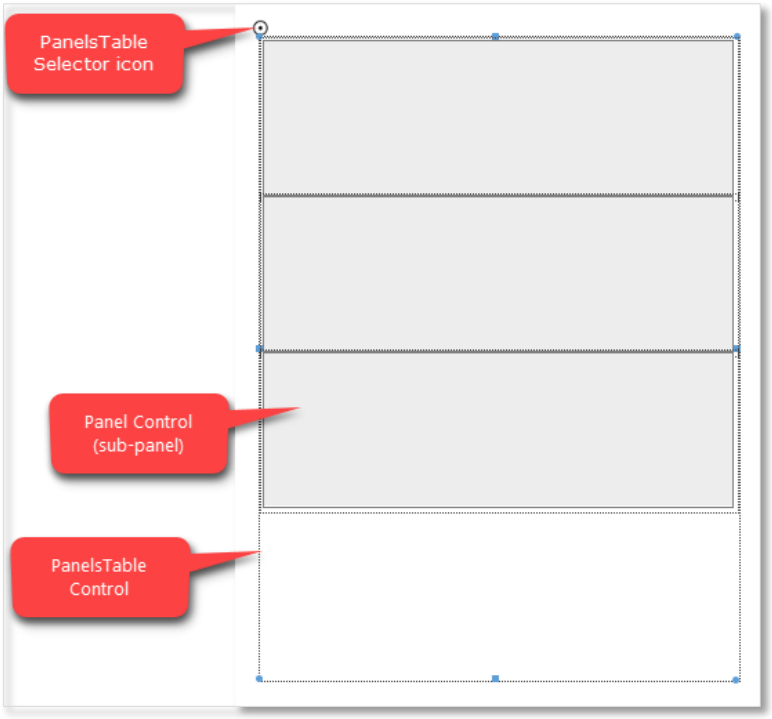
See also

Reference
addAttribute() function
getAttributeValue() function
modifyAttributeValue() function

10.3.4 PanelsTable Control

The **PanelsTable** control is used to control the behavior of different groups of controls inside its panels. The main purpose is to shift controls vertically (up and down) based on their display (showing /hiding).

When a panel inside this control is being hidden, all underneath controls will be shifted up automatically in order to close the gap (spaces) resulted by hiding this control/s and vise versa.



To add a **PanelsTable** control to the form, drag and drop the control onto the form canvas.

You can apply the following operations on the **PanelsTable** control:

- To delete a Panel control from the **PanelsTable** control, select the sub-panel, then press on the Delete key on the keyboard.
- To add a Panel to the **PanelsTable** control, select on the PanelsTable Selector icon ⊙, then click on Add Panel + in the control properties.
- To move a Panel up and down, select the Panel and drag it to the desired location then release the mouse.

Control Properties:
Select the **PanelsTable** control to set the following properties:

- **Control Type:** The type of the selected control.
- **ID:** The Identity of the control. The ID is a unique value and no duplicate IDs allowed in the form.
- **Tooltip:** Appears on the running mode when the user moves the mouse pointer over the control.
- **Auto Form Height:** Set this property to ON in order to control the height of the form automatically when shifting the control up and down. In other words, the height of the form will be adjusted by the height of this control.
- **CSS Classes:** The HTML class attribute makes it possible to define equal styles for elements with the same class name.

- **Style:** The HTML CSS code. The user can edit this code in the editor and change the control style in the designer and art runtime.
- **Add Panel:** Add new sub-panel in the control.
- **Disabled:** Specifies if the control will be disabled or not when the form loads.
- **ReadOnly:** Specifies if the control will be read only or not when the form loads.
- **Hidden:** Specifies if the control will be visible or not when the form loads.

Panel Properties:
Select the **Panel** control to set the following properties (to be applied for each panels in the **PanelsTable** control):

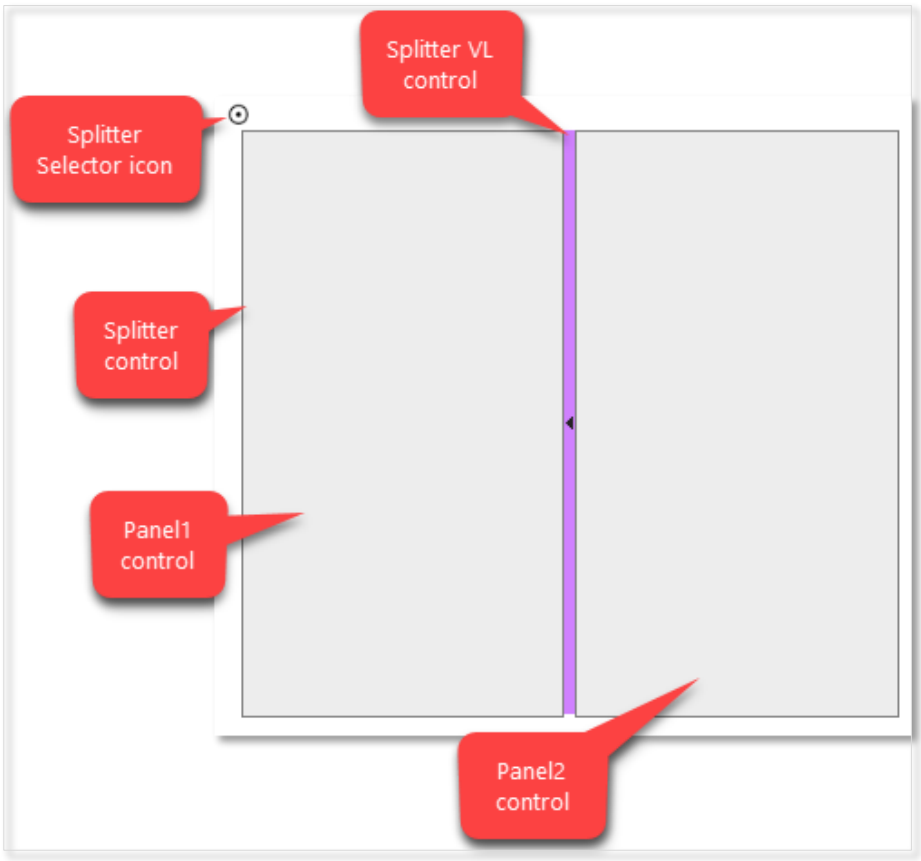
- **Control Type:** The type of the selected control.
- **ID:** The Identity of the control. The ID is a unique value and no duplicate IDs allowed in the form.
- **Tooltip:** Appears on the running mode when the user moves the mouse pointer over the control.
- **Layout:** Please refer to the Layout property in the **Panel** control.
- **CSS Classes:** The HTML class attribute makes it possible to define equal styles for elements with the same class name.
- **Style:** The HTML CSS code. The user can edit this code in the editor and change the control style in the designer and art runtime.
- **Disabled:** Specifies if the control will be disabled or not when the form loads.
- **ReadOnly:** Specifies if the control will be read only or not when the form loads.
- **Hidden:** Specifies if the control will be visible or not when the form loads.

See also

Reference
addAttribute() function
getAttributeValue() function
modifyAttributeValue() function

10.3.5 Splitter Control

The **Splitter** control is a control that represents two panels split by a vertical line (splitter), which allows showing/hiding the left panel when a user clicks on the splitter at run time. You can place any control on the Splitter's control panels.



The designer can change the style of both panels of the **Splitter** control (i.e. resize the panel width) by double clicking the panel or by selecting the panel and press on **Control Style** in the control ribbon. And the same way for changing the style of splitter.

Control Properties:
Select the **Splitter** control to set the following properties:

- **Control Type:** The type of the selected control.
- **ID:** The Identity of the control. The ID is a unique value and no duplicate IDs allowed in the form.
- **Tooltip:** Appears on the running mode when the user moves the mouse pointer over the control.
- **CSS Classes:** The HTML class attribute makes it possible to define equal styles for elements with the same class name.
- **Style:** The HTML CSS code. The user can edit this code in the editor and change the control style in the designer and art runtime.
- **Hidden:** Specifies if the control will be visible or not when the form loads.

Panel Properties:
Select the **Panel** control inside the **Splitter** control to set the following properties (to be applied for both panels in the **Splitter** control):

- **Control Type:** The type of the selected control.
- **ID:** The Identity of the control. The ID is a unique value and no duplicate IDs allowed in the form.

- **Tooltip:** Appears on the running mode when the user moves the mouse pointer over the control.
- **Layout:** Please refer to the Layout property in the **Panel** control.
- **CSS Classes:** The HTML class attribute makes it possible to define equal styles for elements with the same class name.
- **Style:** The HTML CSS code. The user can edit this code in the editor and change the control style in the designer and art runtime.
- **Disabled:** Specifies if the control will be disabled or not when the form loads.
- **ReadOnly:** Specifies if the control will be read only or not when the form loads.
- **Hidden:** Specifies if the control will be visible or not when the form loads.

Splitter VL Properties:
Select the **Splitter VL** control to set the following properties:

- **Control Type:** The type of the selected control.
- **ID:** The Identity of the control. The ID is a unique value and no duplicate IDs allowed in the form.
- **Tooltip:** Appears on the running mode when the user moves the mouse pointer over the control.
- **CSS Classes:** The HTML class attribute makes it possible to define equal styles for elements with the same class name.
- **Style:** The HTML CSS code. The user can edit this code in the editor and change the control style in the designer and art runtime.

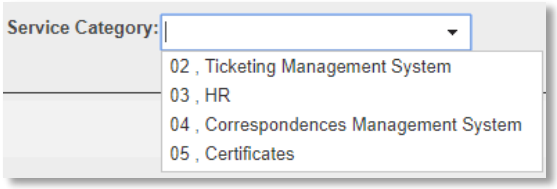
See also

Reference
addAttribute() function
getAttributeValue() function
modifyAttributeValue() function

10.4 Integration Controls

10.4.1 Lookup Control

The **Lookup** control allows user to choose one value or more from a dropdown based on values in a SharePoint list. User can filter the selected values in the Lookup settings based on specific value or another control in the form. This control is searchable to provide user with ability to type inside the dropdown to narrow the results in order to find the required value fast. In the Lookup settings, user can determine which fields to be retrieved in the dropdown, for example, user can retrieve service category and service name as appear in the following screenshot.

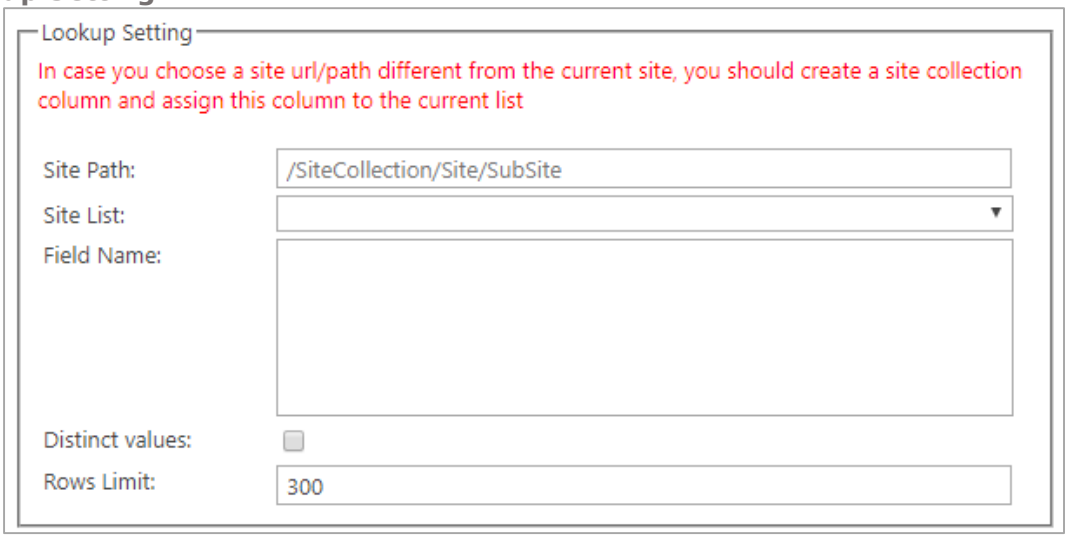


Control Properties:

- **Control Type:** The type of the selected control.
- **ID:** The Identity of the control. The ID is a unique value and no duplicate IDs allowed in the form.
- **Default value:** Set a default value for the control. The value will only be used if no value has been specified in the control rules**.**
- **Tab Index:** Attribute specifies the tab order of an element (when the "tab" button is used for navigating).
- **Placeholder Text:** A short hint that describes the expected value of an input control.
- **Tooltip:** Appears on the running mode when the user moves the mouse pointer over the control.
- **CSS Classes:** The HTML class attribute makes it possible to define equal styles for elements with the same class name.
- **Style:** The HTML CSS code. The user can edit this code in the editor and change the control style in the designer and art runtime.
- **Control Format Type:** Specify how the control will appear at runtime (Lookup, Multiple Values Lookup, CheckBoxes and RadioButtons).
- **Dynamic Data Source**: For more details, Refer to Lookup Selector.
- **Site Link:** The URL of the SharePoint site that contains the source list.
- **Site List:** The list name of the source SharePoint list.
- **Field Name:** The name of the column to show in the lookup control.
- **Field Value:** The name of the column id/value in the lookup control.
- **Required:** Specifies if the control will be required to be filled or not when the form loads. This property will not be active at runtime when both **Required** and **Hidden** properties are set to ON.
- **Required Border:** This property is visible only when the **Required** property is switched ON. This property allows the designer to choose if the red border would appear on the "required" control or not. If not (Required Border is switched OFF), the designer can keep the default border or change it as desired.
- **Disabled:** Specifies if the control will be disabled or not when the form loads.
- **Hidden:** Specifies if the control will be visible or not when the form loads.

Lookup Selector:
When a Lookup control is used, two pieces of data are returned: the ID "value" of the item selected in the lookup and the text "display" of the item selected in the lookup.

Lookup Setting:



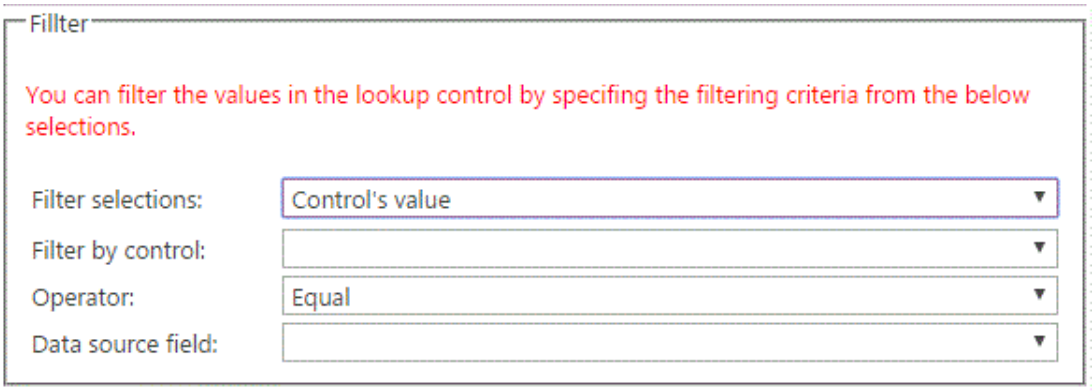
- **Site Path:** The URL of the SharePoint site that contains the source list. This list can be anywhere within the web application. The site picker will only show sites within the current site collection, however, other sites can be entered as a server relative URL manually.
- **Site List:** The list name or ID of the source SharePoint list. The lists available will dynamically be populated based on the specified site. If the system cannot access the site, the list name will need to be specified manually as an expression.
- **Field Name:** The name of the searchable column(s) to show in the lookup control.
- **Distinct values:** If checked, only distinct values of the applicable data will be retrieved.
- **Rows Limit:** The maximum rows of the applicable data to be retrieved. The default value is 300.

Lookup Filter:

You can filter the values in the lookup control by specifying the **Filter selections** from the below selections:

- **No filters:** Values to be shown when there is no value applied to the filter; where nothing is selected in the "filtered by control" or there is no valid specified filter value.
- **Control's value:**
  - **Filter by control:** The control on the current form to filter the available items by.
  - **Operator:** The operation type (Equal, Not equal, Greater than or equal to, Greater than, ...)
  - **Data source field:** The field in the source list to apply the filter to.

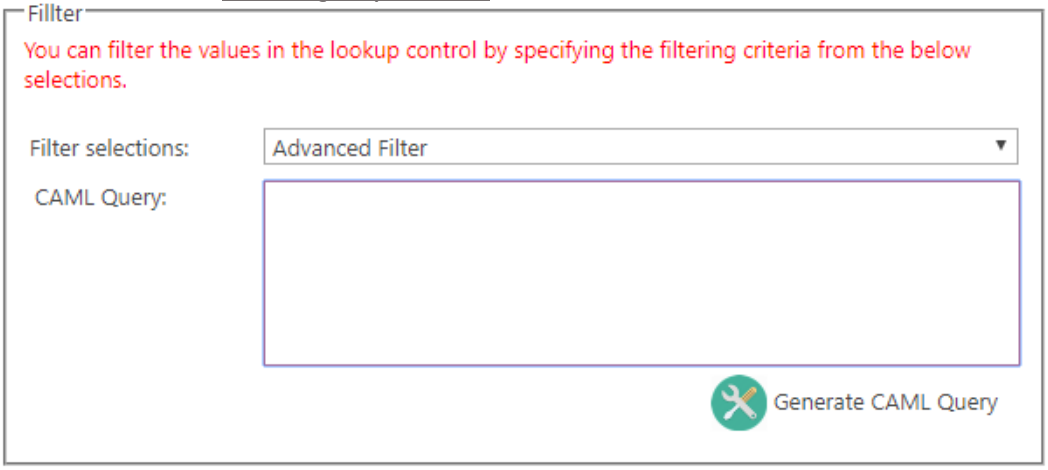
  Note: The filter doesn't work with other multi-select lookup controls; in this case it is better to change them to checkboxes or dropdown list.



- **Specific value:**
  - **Filter by this value:** The value to filter the available items by.
  - **Operator:** the operation type (Equal, Not equal, Greater than or equal to, Greater than, ...)
  - **Data source field:** Specify the CAML query The field in the source list to apply the filter to.

- **Advanced Filter:**
  - **CAML Query:** In this area you can filter the returned values by writing a CAML query or by clicking on "Generate CAML Query" that will help you create the query. Once you finish creating the query, click on the Insert button on the CAML Query Builder menu to insert the created query in the CAML Query area in the Advanced Filter.

    You can refer to CAML Query Builder for more details.



See also

Reference
clear() function
getLookupSelectedText() function
isEmpty() function
isEqual() function
isGreaterEqualThan() function
isGreaterThan() function
isNotEmpty() function
isNotEqual() function
isNumber() function
isSmallerEqualThan() function
isSmallerThan() function
length() function
math() function

not() function
notContain() function
notEndWith() function
notStartWith() function
getValue() function
setValue() function
max() function
min() function
setFocus() function
split() function
startWith() function
substring() function
sum() function
toLower() function
toUpper() function
toUpperFirst() function
Trim() function
addAttribute() function
getAttributeValue() function
modifyAttributeValue() function
show() function

10.4.2 Advanced Lookup Control

Note: This topic applies to SPARK Reports Builder Standard and Enterprise Editions only.

The Advanced Lookup control allows users to make selections based on values in a SharePoint list. The lookup data are not loaded during form load, so this control is the most suitable to display data from large lists. It allows users to search for the required value using a popup search dialog, activated by the "Selector" button. The Advanced Lookup control can be configured to consume data from any list within the local site collection or from other site collections in the web application.

When dragging the Advanced Lookup control to the form design area and clicking on the icon beside the textbox of the control "Selector Button", a search dialog appears in order to enable the user to search for any data in the source list, which is specified in the "**Lookup Field Selector**" data source during the control configuration in the design mode, the user will be able to sort the results ascending or descending per column by clicking on any column header in the search dialog. The selected value will be presented in the form as a link to redirect the user to the selected item's view (display) page when clicking on that link in the form.

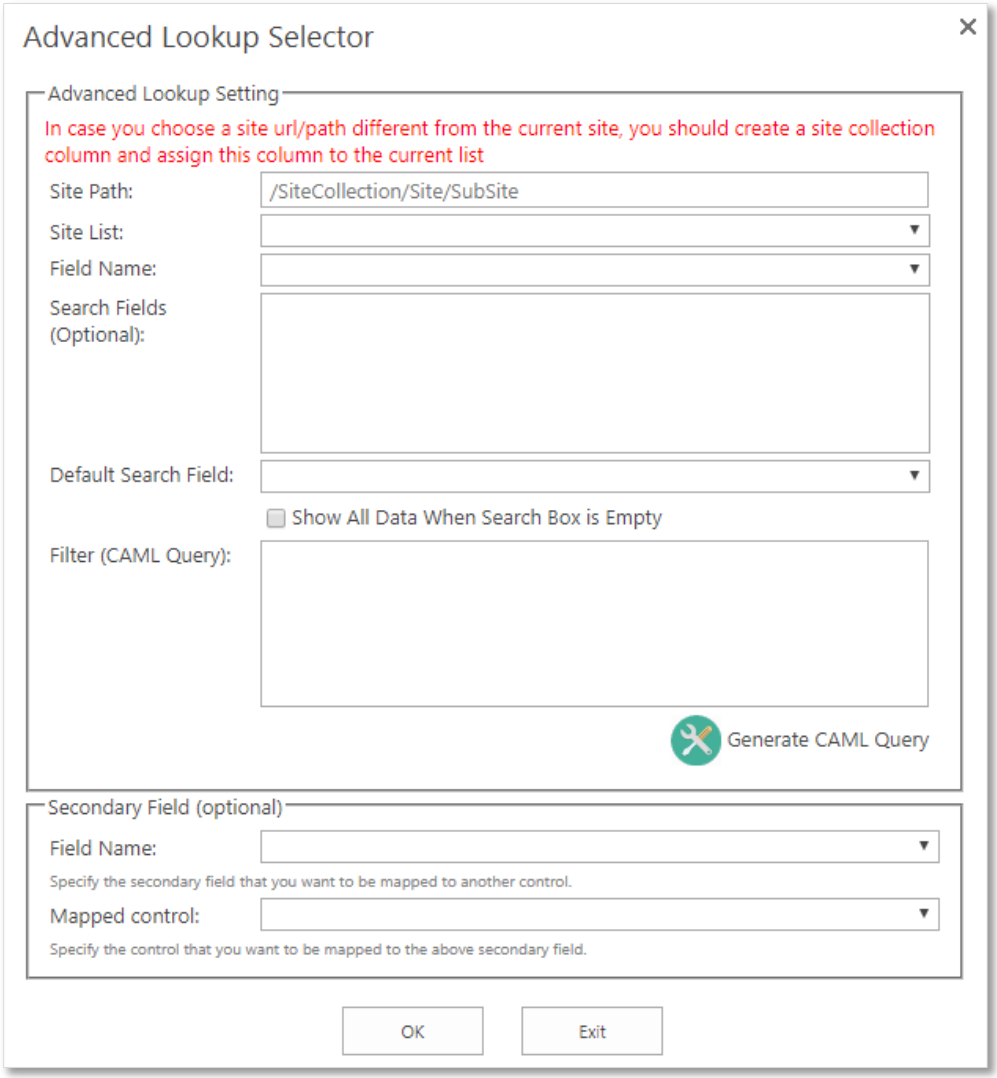
In the "Field Selector" of the **Advanced Lookup** you can bind two fields in a form with the corresponding one in the list. The first field is the field to be shown on the Ad-Hoc form when you select the value from the search dialog; the second one (optional) is any field in the list that you can use its value in rules and functions as well.

Control Properties:

- **Control Type:** The type of the selected control.
- **ID:** The Identity of the control. The ID is a unique value and no duplicate IDs allowed in the form.
- **Tab Index:** Attribute specifies the tab order of an element (when the "tab" button is used for navigating).
- **Tooltip:** Appears on the running mode when the user moves the mouse pointer over the control.

- **CSS Classes:** The HTML class attribute makes it possible to define equal styles for elements with the same class name.
- **Style:** The HTML CSS code. The user can edit this code in the editor and change the control style in the designer and art runtime.
- **Dynamic Data Source:** For more details, refer to Advanced Lookup Selector.
- **Site Link:** The URL of the SharePoint site that contains the source list.
- **Site List:** The list name of the source SharePoint list.
- **Field Name:** The name of the column to show in the lookup control.
- **Field Value:** The name of the column id/value in the lookup control.
- **Dialog Width:** Specifies the width of the control's search/select dialog.
- **Dialog Height:** Specifies the height of the control's search/select dialog.
- **Header BG Color:** Specifies the color of the header's background in the control's search dialog.
- **Dialog Border Color:** Specifies the color of the border in the control's search dialog.
- **Dialog Border Size:** Specifies the size of the border in the control's search dialog.
- **Dialog Border Style:** Specifies the style of the border in the control's search dialog.
- **Required:** Specifies if the control will be required to be filled or not when the form loads. This property will not be active at runtime when both **Required** and **Hidden** properties are set to ON.
- **Required Border:** This property is visible only when the **Required** property is switched ON. This property allows the designer to choose if the red border would appear on the "required" control or not. If not (Required Border is switched OFF), the designer can keep the default border or change it as desired.
- **Disabled:** Specifies if the control will be disabled or not when the form loads.
- **ReadOnly:** Specifies if the control will be read only or not when the form loads.
- **Hidden:** Specifies if the control will be visible or not when the form loads.

Advanced Lookup Selector:



When Advanced Lookup control is used, two pieces of data are returned: the ID of the item selected in the lookup and the text of the item selected in the lookup. The "Field Name" property in the setting dialog is bind to the field text "display" value while the field value is the ID referring to the selected list item in the search dialog.

Advanced Lookup Setting:

- **Site Path:** The URL of the SharePoint site that contains the source list. This list can be anywhere within the web application. The site picker will only show sites within the current site collection, however, other sites can be entered as a server relative URL manually.
- **Site List:** The list name or ID of the source SharePoint list. The lists available will be dynamically populated based on the specified site. If the system cannot access the site, the list name will need to be specified manually as an expression.
- **Field Name:** The name of the column to show in the advanced lookup control.
- **Search Fields** (Optional): Specify the fields to be shown, at run-time, in the "Search for" dropdown in the "Search Dialog" popup, so the user can select from one of the specified fields. If this attribute left empty, the system will return all list columns in the "Search for" dropdown.
- **Default Search Field:** Specify the default search field to be shown at run-time in the "Search for" dropdown in the "Search Dialog" popup.
- **Show All Data When Search Box is Empty:** If checked, the applicable data will be retrieved automatically when open the search dialog without needs to specify the value in the search box.

- **Filter (CAML Query):** In this property, you can filter the returned values by writing a CAML query or by clicking on "Generate CAML Query" that will help you create the query. Once you finish creating the query, click on the Insert button on the CAML Query Builder menu to insert the created query in the CAML Query area in the Filter.

  You can refer to CAML Query Builder for more details.

Secondary Field (Optional):

- **Field Name:** The secondary field name that you want to be mapped to another control.
- **Mapped Control:** The control on the current form to be mapped to the above secondary field.

See also

Reference
clear() function
getAdvancedLookupData() function
generateAdvancedLookupData() function
getValue() function
setValue() function
setFocus() function
addAttribute() function
getAttributeValue() function
modifyAttributeValue() function

10.4.3 People Picker Control

The **People Picker** control allows users to find and select others users and groups in the SP environment or DC LDAP, you can filter and restrict the results that are displayed when a user searches for a user or group.

Control Properties:

- **Control Type:** The type of the selected control.
- **ID:** The Identity of the control. The ID is a unique value and no duplicate IDs allowed in the form.
- **Tab Index:** Attribute specifies the tab order of an element (when the "tab" button is used for navigating).
- **Tooltip:** Appears on the running mode when the user moves the mouse pointer over the control.
- **CSS Classes:** The HTML class attribute makes it possible to define equal styles for elements with the same class name.
- **Style:** The HTML CSS code. The user can edit this code in the editor and change the control style in the designer and art runtime.
- **Multiple Selections**: Allow multiple people/groups to be selected.
- **Selection of:** To choose the user entity type that can be specified in the control, there are two options available (people or people and groups).
- **Choose from:** Attribute specifies the people picker control's users pool, the designer can choose (All Users) option and in this case the users container will be all users in the current site collection, or the designer can choose any SharePoint group in the dropdown in order to make the users container assigned to this group only.
- **Required:** Specifies if the control will be required to be filled or not when the form loads. This property will not be active at runtime when both **Required** and **Hidden** properties are set to ON.
- **Required Border:** This property is visible only when the **Required** property is switched ON. This property allows the designer to choose if the red border would appear on the "required" control or not. If not (Required Border is switched OFF), the designer can keep the default border or change it as desired.
- **Disabled:** Specifies if the control will be disabled or not when the form loads.
- **ReadOnly:** Specifies if the control will be read only or not when the form loads.
- **Hidden:** Specifies if the control will be visible or not when the form loads.

See also

Reference
clear() function
setPeoplePicker() function
getPeoplePickerValue() function
getValue() function
getSiteCollectionUserID() function
setFocus() function
addAttribute() function
getAttributeValue() function
modifyAttributeValue() function

10.4.4 External Data Dialog Control

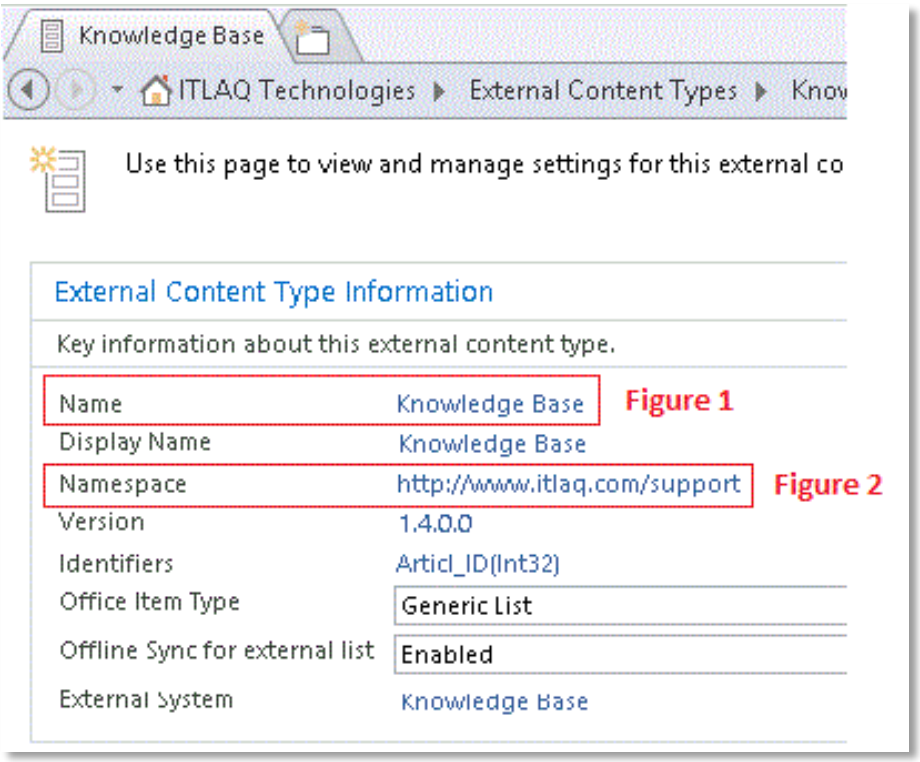
Note: This topic applies to SPARK Reports Builder Standard and Enterprise Editions only.

External Data Dialog control is specifically for consuming data from BCS. When added to a form, this control provides an easy way for users to select items from the data source to which BCS is connecting. For example, perhaps the user is filling out a

customer service form, and he likes to select the customer name directly from BCS rather than creating a duplicate list of that information in SharePoint.

Control Properties:

- **Control Type:** The type of the selected control.
- **ID:** The Identity of the control. The ID is a unique value and no duplicate IDs allowed in the form.
- **Tab Index:** Attribute specifies the tab order of an element (when the "tab" button is used for navigating).
- **Tooltip:** Appears on the running mode when the user moves the mouse pointer over the control.
- **CSS Classes:** The HTML class attribute makes it possible to define equal styles for elements with the same class name.
- **Style:** The HTML CSS code. The user can edit this code in the editor and change the control style in the designer and art runtime.
- **ECT NameSpace:** External content type namespace, see (Figure 2) in the SharePoint designer page.
- **ECT Name:** External content type name, see (Figure 1) in the SharePoint designer page.
- **System Instance Name:** The system instance name is the (External System) in the SharePoint designer ECT creation page (Figure 3).
- **Display Field Name:** Specify the field that you need to be shown in the control when the user selects from the dialog.
- **EDF Name:** External data field Name that specifies the external content type field, which you want to be mapped to another control, you can think about this property as an extension of driven data from the selected row to be mapped to other controls in the form (usually TextBox, TextArea or label).
- **EDF mapped control:** External data field mapped control specifies the control that you want to set the EDF value in, this value has to be specified in the (EDF Name) property.
- **ECT Finder view:** External content type finder view is usually used in case there is no default ECT Finder Method Instance name for this instance, this property is (Optional).
- **Dialog Width:** Specifies the width of the control’s search/select dialog.
- **Dialog Height:** Specifies the height of the control’s search/select dialog.
- **Header BG Color:** Specifies the color of the header’s background in the control’s search dialog.
- **Dialog Border Color:** Specifies the color of the border in the control’s search dialog.
- **Dialog Border Size:** Specifies the size of the border in the control’s search dialog.
- **Dialog Border Style:** Specifies the style of the border in the control’s search dialog.
- **Required:** Specifies if the control will be required to be filled or not when the form loads. This property will not be active at runtime when both **Required** and **Hidden** properties are set to ON.
- **Required Border:** This property is visible only when the **Required** property is switched ON. This property allows the designer to choose if the red border would appear on the "required" control or not. If not (Required Border is switched OFF), the designer can keep the default border or change it as desired.
- **Disabled:** Specifies if the control will be disabled or not when the form loads.
- **ReadOnly:** Specifies if the control will be read only or not when the form loads.
- **Hidden:** Specifies if the control will be visible or not when the form loads.



See also

Reference
clear() function
getValue() function
setValue() function
setFocus() function
getECTData() function
generateECTData() function
addAttribute() function
getAttributeValue() function
modifyAttributeValue() function

10.4.5 SQL Connector Control

Note: This topic applies to SPARK Reports Builder Enterprise Edition only.

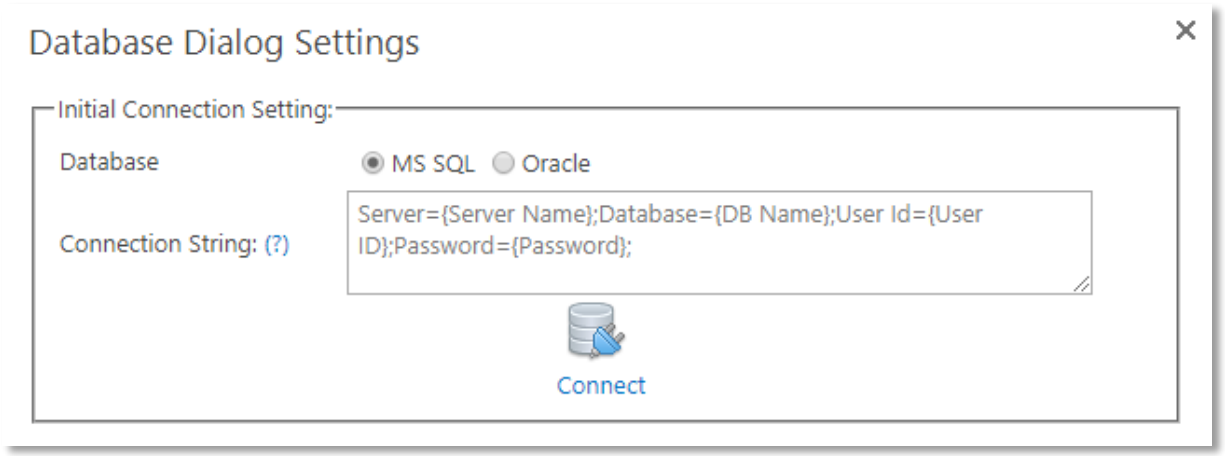
The **SQL Connector** control allows users to make selections based on values in a SQL database. The SQL Connector control returns a primary and secondary values based on the selected **Fields Name** specified in the **Database Dialog Setting**. It can also consume all the selected fields values returned from the search dialog using control's rules.

Control Properties:

- **Control Type:** The type of the selected control.
- **ID:** The Identity of the control. The ID is a unique value and no duplicate IDs allowed in the form.
- **Tab Index:** Attribute specifies the tab order of an element (when the "tab" button is used for navigating).
- **Tooltip:** Appears on the running mode when the user moves the mouse pointer over the control.

- **CSS Classes:** The HTML class attribute makes it possible to define equal styles for elements with the same class name.
- **Style**: The HTML CSS code. The user can edit this code in the editor and change the control style in the designer and art runtime.
- **Control Settings:** For more details, refer to Database Dialog Setting.
- **Dialog Width:** Specifies the width of the control's search/select dialog.
- **Dialog Height:** Specifies the height of the control's search/select dialog.
- **Header BG Color:** Specifies the color of the header's background in the control's search dialog.
- **Dialog Border Color:** Specifies the color of the border in the control's search dialog.
- **Dialog Border Size:** Specifies the size of the border in the control's search dialog.
- **Dialog Border Style:** Specifies the style of the border in the control's search dialog.
- **Required:** Specifies if the control will be required to be filled or not when the form loads. This property will not be active at runtime when both **Required** and **Hidden** properties are set to ON.
- **Required Border:** This property is visible only when the **Required** property is switched ON. This property allows the designer to choose if the red border would appear on the "required" control or not. If not (Required Border is switched OFF), the designer can keep the default border or change it as desired.
- **Disabled:** Specifies if the control will be disabled or not when the form loads.
- **ReadOnly:** Specifies if the control will be read only or not when the form loads.
- **Hidden:** Specifies if the control will be visible or not when the form loads.

Database Dialog Setting:



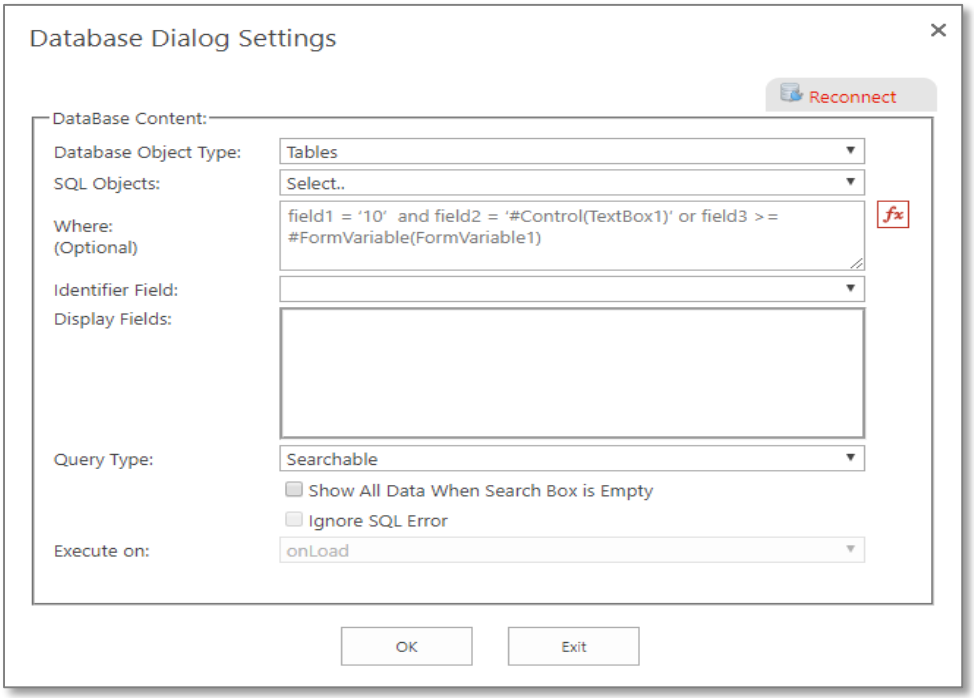
- **Database:** Specifies the database type to connect to [SQL/Oracle].
- **Connection String:** The Connection String of the SQL Database that you want to connect with. In this case "SQL Server Database".

This is an example of a Connection String to connect to SQL Server DB: Server={Server Name};Database={DB Name};User Id={User ID}; Password={Password};

Note: The Database Object Type will return nothing if the Connection String is incorrect.

Database Content:

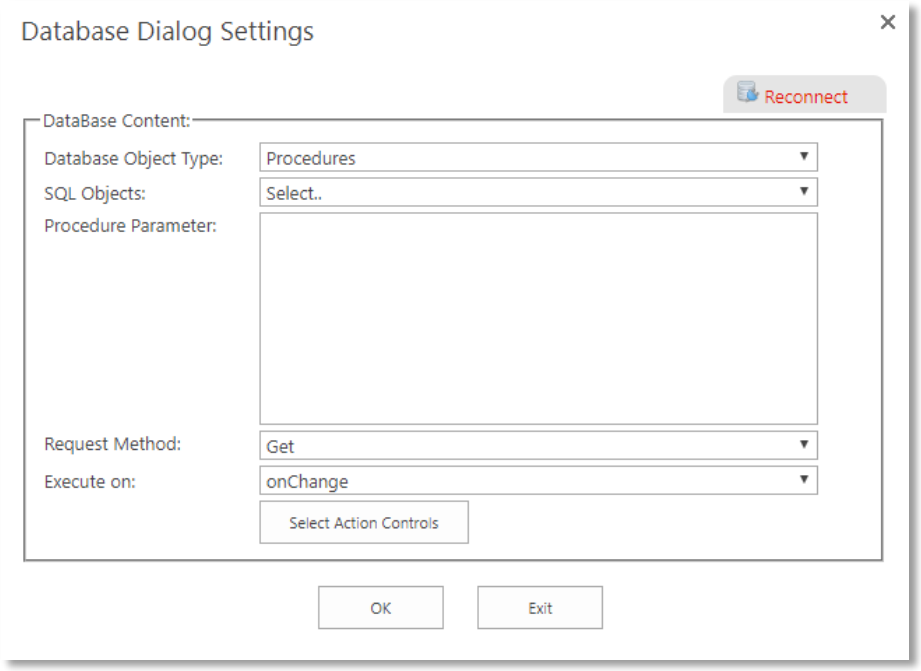
- **Database Object Type:** Select Tables, Views or Procedures of the provided DB to be retrieved.
- **For Tables or Views:**



- **SQL Objects:** Returns all tables/views in the database.
- **Where:** (optional) used to filter records in the selected object. You can use static values, form variables and other form fields in the where. Example: field1 = '10' and field2 = '#Control(TextBox1)' or field3 >= #FormVariable(FormVariable1). You can add input values (form controls, form variables or static values) by clicking on *fx*.
- **Identifier Field:** The field name that you want to be mapped to the control.
- **Display Fields:** If "Searchable" is selected as a "Query Type", the system will return the selected fields in the "Search for" dropdown in the "Search Dialog" popup at run-time, so the user can select from one of the specified fields to search for. If this attribute left empty, the system will return what is specified in the "Identifier Field". You can use the Display Fields in form variables, rules and functions.
- **Query Type:** Defines the control type you would like to represent the SQL data within, the types are: Searchable, Checkbox List, Dropdown List or Radio Button.

  Note: The system will automatically create a new rule when the selected query type is Checkbox List, Dropdown List or Radio Button.
- **Show All Data When Search Box is Empty:** If checked, the applicable data will be retrieved automatically when open the search dialog in the run-mode without needs to specify the value in the search box (text-search criteria).
- **Ignore SQL Error:** Ignore any SQL error on runtime.
- **Execute on:** Defines the way this control executes the SQL query through, the execution trigger could be onLoad or onChange.
- **Select Action Controls:** Appears only when you select the onChange (execute on) option, when clicking on this button it will show a dialog containing all form controls that can trigger the rule executing the SQL query. In general, you can select the controls you want to update the query's data based on their actions (Dynamic Query).

- **For Procedures:**



- **SQL Objects:** Returns all procedures in the database.
- **Procedure Parameter:** Returns all procedure's parameters if exist. You can add input values (form controls, form variables and static values) by clicking on fx.
- **Request Method:** Specifies the Request Method: GET or POST. GET is to return values from the specified procedure and POST is to insert, update or delete record(s) based on the design of that specified procedure. In case the designer selected the "GET" method, the retrieved data will be stored in the SQL Connector control as a JSON string, and these data can be consumed and populated on the different form's controls by using the getSQLProcedureData() internal function.
- **Execute on:** Defines the way this control executes the SQL query through, the execution trigger could be onLoad or onChange.
- **Select Action Controls:** Appears only when you select the onChange (execute on) option, when clicking on this button it will show a dialog containing all form controls that can trigger the rule executing the SQL query. In general, you can select the controls you want to update the query's data based on their actions (Dynamic Query).

See also

Reference
clear() function
getSQLData() function
generateSQLData() function
getValue() function
addAttribute() function
getAttributeValue() function
modifyAttributeValue() function
getSQLProcedureData() function
executeSQLQuery() function

10.4.6 XML Connector Control

Note: This topic applies to SPARK Reports Builder Enterprise Edition only.

The **XML Connector** control allows users to integrate the form with an external or internal XML data source, XML string stored in SP columns or InfoPath forms.

Control Properties:

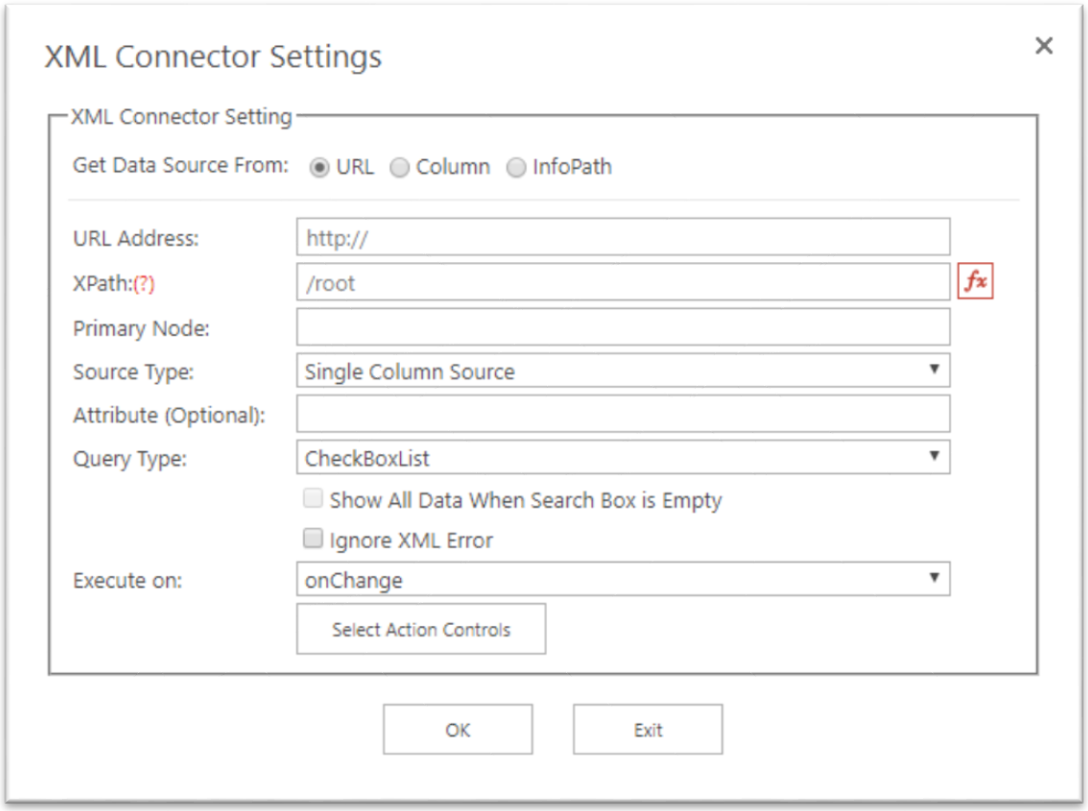
- **Control Type:** The type of the selected control.
- **ID:** The Identity of the control. The ID is a unique value and no duplicate IDs allowed in the form.
- **Tab Index:** Attribute specifies the tab order of an element (when the "tab" button is used for navigating).
- **Tooltip:** Appears on the running mode when the user moves the mouse pointer over the control.
- **CSS Classes:** The HTML class attribute makes it possible to define equal styles for elements with the same class name.
- **Style:** The HTML CSS code. The user can edit this code in the editor and change the control style in the designer and art runtime.
- **Control Settings:** For more details, click on XML Dialog Setting.
- **Dialog Width:** Specifies the width of the control's search/select dialog.
- **Dialog Height:** Specifies the height of the control's search/select dialog.
- **Header BG Color:** Specifies the color of the header's background in the control's search dialog.
- **Dialog Border Color:** Specifies the color of the border in the control's search dialog.
- **Dialog Border Size:** Specifies the size of the border in the control's search dialog.
- **Dialog Border Style:** Specifies the style of the border in the control's search dialog.
- **Required:** Specifies if the control will be required to be filled or not when the form loads. This property will not be active at runtime when both **Required** and **Hidden** properties are set to ON.
- **Required Border:** This property is visible only when the **Required** property is switched ON. This property allows the designer to choose if the red border would appear on the "required" control or not. If not (Required Border is switched OFF), the designer can keep the default border or change it as desired.
- **Disabled:** Specifies if the control will be disabled or not when the form loads.
- **ReadOnly:** Specifies if the control will be read only or not when the form loads.
- **Hidden:** Specifies if the control will be visible or not when the form loads.

XML Dialog Setting:

You can connect to the XML data source by specifying the following:

1. An external or internal URL that the XML stored in;
2. A column string stored in a SharePoint list with an XML structure;
3. An InfoPath item stored in a SharePoint list.

URL Settings:



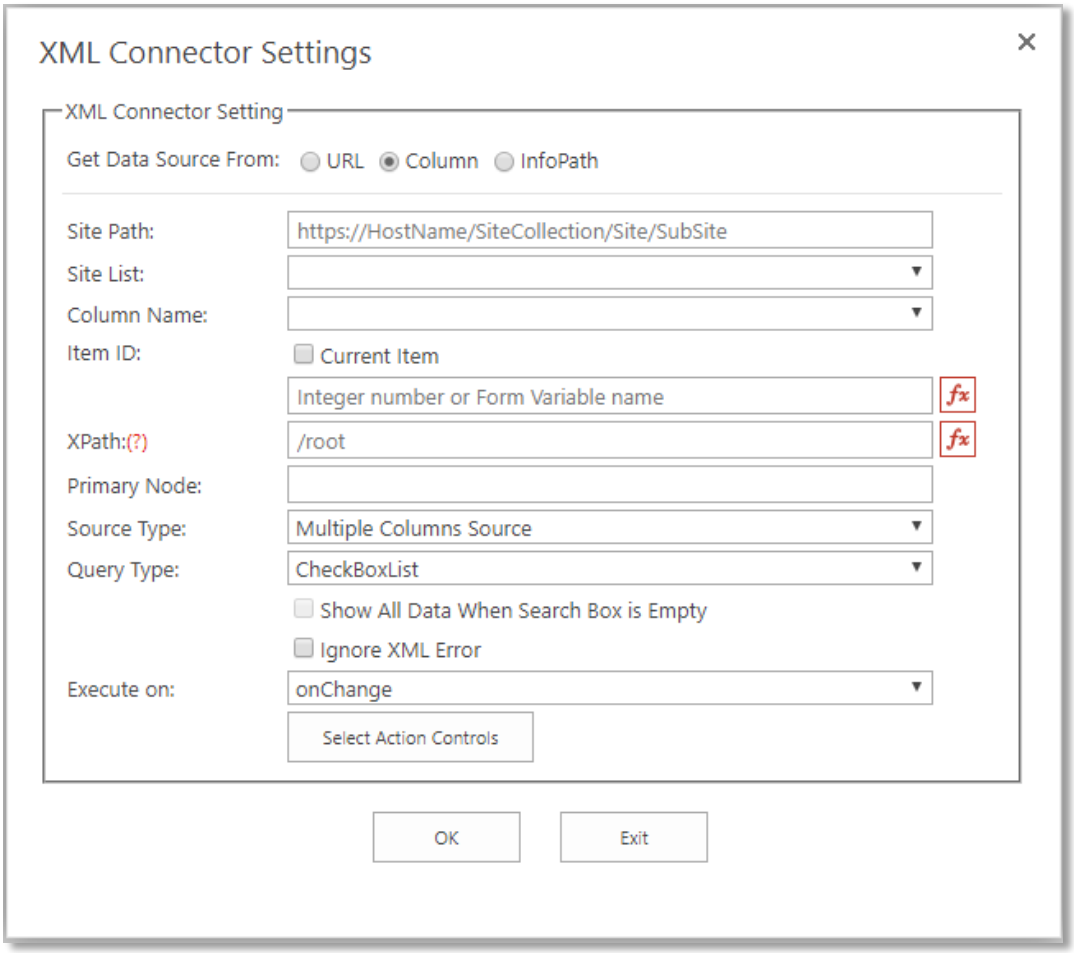
- **URL Address:** Specify the URL in which the XML stored.
- **XPath:** Specify the XPath of the XML. The system returns the root path automatically and you can change it. Note: The XPath can contain static values, form variables or form's controls values to reform a dynamic filter, for example:
    - XPath: //title[@lang='en']
      Selects all the title elements that have a "lang" attribute with a value of "en".
    - XPath: //title[@lang='**#Control(TextBox1)']**
      Selects all the title elements that have a "lang" attribute with a value of the control name TextBox1.
    - //title[@lang='**#FormVariable(FormVariable1)']**
      Selects all the title elements that have a "lang" attribute with a value of the Form Variable name FormVariable1.

      Note: You can add input values (form controls, form variables and static values) by clicking on *fx*.
- **Primary Node:** The system automatically returns all nodes of the XPath. Select the Node that you want to retrieve data in the form based on this input.
- **Source Type:** Presents the XML structure type, we have two types of XML: structure Multiple Columns Source and Single Column Source. The multiple column source will translate the XML structure into a multiple column in the search dialog, while the single column source will translate the xml structure into a single column in the search dialog, the designer is responsible in choosing the right type for his XML data source structure.
- **Attribute (Optional):** Available only when the designer selects the single column source from the source type and used to return all the nodes that contain a specific Attribute name, the attribute value will be searchable in the search dialog.
- **Query Type:** Defines the control type you would like to represent the XML data within, the types are: Searchable, Checkbox List, Dropdown List or Radio Button.

Note: The system will automatically create a new rule when the selected query type is Checkbox List, Dropdown List or Radio Button.

- **Show All Data When Search Box is Empty:** If checked, the applicable data will be retrieved automatically when open the search dialog in the run-mode without needs to specify the value in the search box (text-search criteria).
- **Ignore XML Error:** Ignore any XML error on runtime.
- **Execute on:** Defines the way this control executes the XPath of the XML through. The execution trigger could be onLoad or onChange.
- **Select Action Controls:** Appears only when you select the onChange (execute on) option, when clicking on this button it will show a dialog containing all form controls that can trigger the rule executing the XPath of the XML. In general, you can select the controls you want to update the query's data based on their actions (Dynamic Query).

Column Settings:



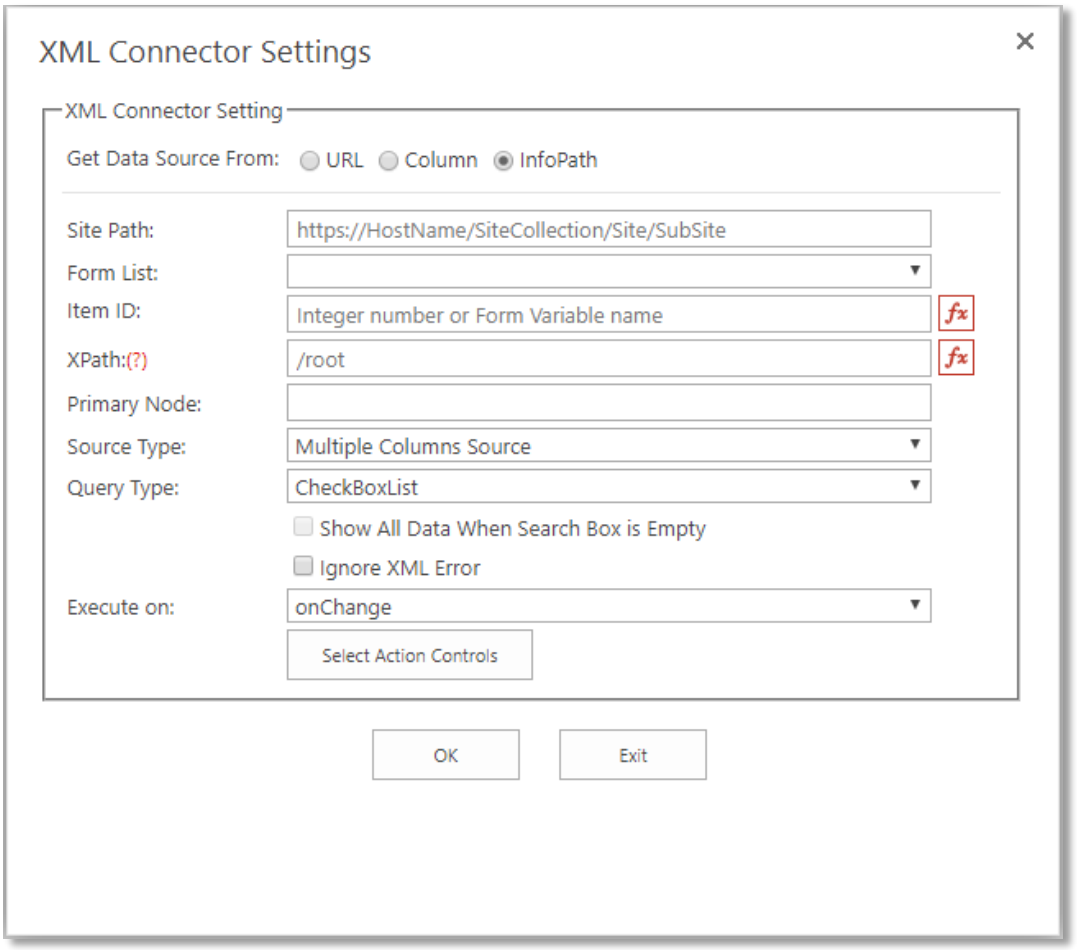
- **Site Path:** Specify the Site in which the XML stored.
- **Site List:** Choose the List in which the XML stored.
- **Column Name:** Choose the XML string column.
- **Item ID:** Specify the Item ID. You can specify the input value of the Item ID by clicking on *fx*, the input value can be form controls, form variables or static values. The Item ID is visible if the Current Item checkbox is not checked. Please note that you can only read XML data from specific item in the list. If you check "Current Item", the system will read from the current item of the current list so no need to specify the Item ID.
- **XPath:** Specify the XPath of the XML. The system returns the root path automatically and you can change it. Note: The XPath can contain static values, form variables or form controls values to reform a dynamic filter, for example:

a. XPath: //title[@lang='en']
   Selects all the title elements that have a "lang" attribute with a value of "en".
b. XPath: //title[@lang='**#Control(TextBox1)']**
   Selects all the title elements that have a "lang" attribute with a value of the control name TextBox1.
c. //title[@lang='**#FormVariable(FormVariable1)']**
   Selects all the title elements that have a "lang" attribute with a value of the Form Variable name FormVariable1.

Note: You can add input values (form controls, form variables and static values) by clicking on *fx*.

- **Primary Node:** Select the Primary Node that you want to retrieve data in the form based on this input.
- **Source Type:** Presents the XML structure type, we have two types of XML: structure Multiple Columns Source and Single Column Source. The **Multiple** columns' source will translate the XML structure into a multiple column in the search dialog, while the **Single** column source will translate the xml structure into a single column in the search dialog. The designer is responsible for choosing the right type for his XML data source structure.
- **Attribute (Optional):** Available only when the designer selects the single column source from the source type and used to return all the nodes that contain a specific Attribute name, the attribute value will be searchable in the search dialog.
- **Query Type:** Defines the control type you would like to represent the XML data within, the types are: Searchable, Checkbox List, Dropdown List or Radio Button.
  Note: The system will automatically create a new rule when the selected query type is Checkbox List, Dropdown List or Radio Button.
- **Show All Data When Search Box is Empty:** If checked, the applicable data will be retrieved automatically when open the search dialog in the run-mode without needs to specify the value in the search box (text-search criteria).
- **Ignore XML Error:** Ignore any XML error on runtime.
- **Execute on:** Defines the way this control executes the XPath of the XML through. The execution trigger could be onLoad or onChange.
- **Select Action Controls:** Appears only when you select the onChange (execute on) option, when clicking on this button it will show a dialog containing all form controls that can trigger the rule executing the XPath of the XML. In general, you can select the controls you want to update the query's data based on their actions (Dynamic Query).

InfoPath Settings:



- **Site Path:** Specify the Site in which the XML stored.
- **Form List:** Choose the List in which the XML stored. Please note that the list should be of the Form type.
- **Item ID:** Specify the Item ID. You can specify the input value of the Item ID by clicking on *fx*, the input value can be form controls, form variables or static values. Please note that you can only read XML data from specific item in the list.
- **XPath:** Specify the XPath of the XML. The system returns the root path automatically and you can change it. Note: The XPath can contain static values, form variables or form controls values to reform a dynamic filter, for example:
    a. XPath: //title[@lang='en']
    Selects all the title elements that have a "lang" attribute with a value of "en".
    b. XPath: //title[@lang='**#Control(TextBox1)'**]
    Selects all the title elements that have a "lang" attribute with a value of the control name TextBox1.
    c. //title[@lang='**#FormVariable(FormVariable1)'**]
    Selects all the title elements that have a "lang" attribute with a value of the Form Variable name FormVariable1.

  Note: You can add input values (form controls, form variables and static values) by clicking on *fx*.

- **Primary Node:** Select the Primary Node that you want to retrieve data in the form based on it.
- **Source Type:** Presents the XML structure type, we have two types of XML: structure Multiple Columns Source and Single Column Source. The multiple column source will translate the XML structure into a multiple column in the search dialog, while the single column source will translate the xml structure into

a single column in the search dialog, the designer is responsible in choosing the right type for his XML data source structure.

- **Attribute (Optional):** Available only when the designer selects the single column source from the source type and used to return all the nodes that contain a specific Attribute name, the attribute value will be searchable in the search dialog.
- **Query Type:** Defines the control type you would like to represent the XML data within, the types are: Searchable, Checkbox List, Dropdown List or Radio Button.
  Note: The system will automatically create a new rule when the selected query type is Checkbox List, Dropdown List or Radio Button.
- **Show All Data When Search Box is Empty:** If checked, the applicable data will be retrieved automatically when open the search dialog in the run-mode without needs to specify the value in the search box (text-search criteria).
- **Ignore XML Error:** Ignore any XML error on runtime.
- **Execute on:** Defines the way this control executes the XPath of the XML through. The execution trigger could be onLoad or onChange.
- **Select Action Controls:** Appears only when you select the onChange (execute on) option, when clicking on this button it will show a dialog containing all form controls that can trigger the rule executing the XPath of the XML. In general, you can select the controls you want to update the query's data based on their actions (Dynamic Query).

See also

Reference
clear() function
getXMLData() function
generateXMLData() function
executeXMLQuery() function
getValue() function
setFocus() function
addAttribute() function
getAttributeValue() function
modifyAttributeValue() function

10.4.7 Web Connector Control

Note: This topic applies to SPARK Reports Builder Enterprise Edition only.

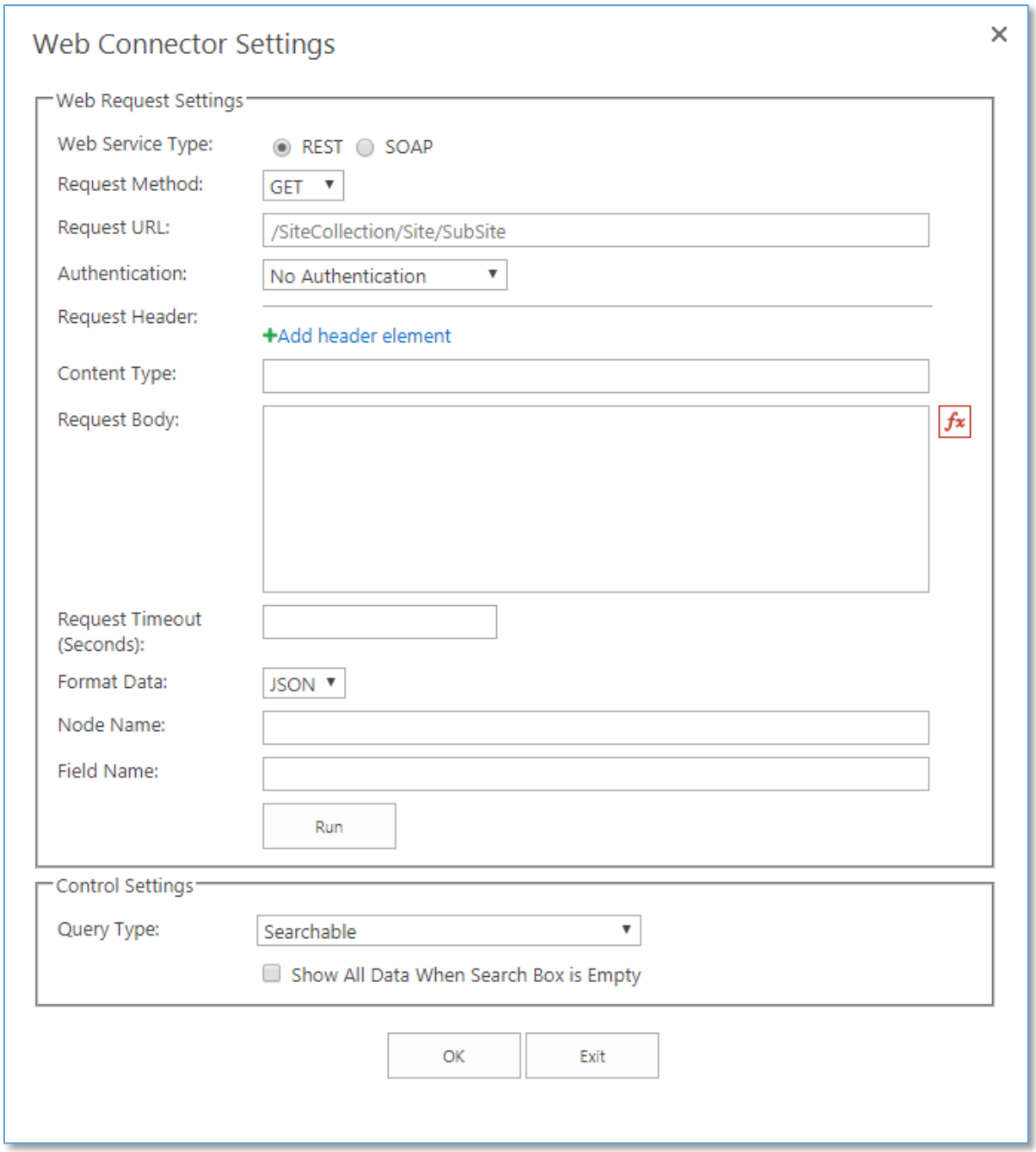
The **Web Connector** control allows users to integrate the form with a Web Service data source using REST or SOAP methods.

Control Properties:

- **Control Type:** The type of the selected control.
- **ID:** The Identity of the control. The ID is a unique value and no duplicate IDs allowed in the form.
- **Tab Index:** Attribute specifies the tab order of an element (when the "tab" button is used for navigating).
- **Tooltip:** Appears on the running mode when the user moves the mouse pointer over the control.
- **CSS Classes:** The HTML class attribute makes it possible to define equal styles for elements with the same class name.
- **Style:** The HTML CSS code. The user can edit this code in the editor and change the control style in the designer and art runtime.
- **Control Settings:** For more details, refert to Web Connector Setting.
- **Dialog Width:** Specifies the width of the control's search/select dialog.
- **Dialog Height:** Specifies the height of the control's search/select dialog.

- **Header BG Color:** Specifies the color of the header's background in the control's search dialog.
- **Dialog Border Color:** Specifies the color of the border in the control's search dialog.
- **Dialog Border Size:** Specifies the size of the border in the control's search dialog.
- **Dialog Border Style:** Specifies the style of the border in the control's search dialog.
- **Required:** Specifies if the control will be required to be filled or not when the form loads. This property will not be active at runtime when both **Required** and **Hidden** properties are set to ON.
- **Required Border:** This property is visible only when the **Required** property is switched ON. This property allows the designer to choose if the red border would appear on the "required" control or not. If not (Required Border is switched OFF), the designer can keep the default border or change it as desired.
- **Disabled:** Specifies if the control will be disabled or not when the form loads.
- **ReadOnly:** Specifies if the control will be read only or not when the form loads.
- **Hidden:** Specifies if the control will be visible or not when the form loads.

Web Connector Settings:



- **Web Service Type:** Specify the web service type (REST/SOAP).
- **Request Method:** Specify the web request method (GET, PUT or POST)
- **Request URL:** Specify the URL in which the web service located.
- **Authentication:** Specify the authentication method: No Authentication, Windows Authentication, Forms Authentication or Impersonate Identity. In case of selecting the Form Authentication option, an icon will display next to the property, when clicking on it you need to provide access account credentials (**Username and Password**), these information will be encrypted and securely stored within the system. If Impersonate Identity option is selected then the "Secure Store Application ID" property will display down the property, you need to insert the secure store identity you have created in the SharePoint Secure store service in order to allow the Web Connector Control to access the web services specified in a highly secured way.

- **Request Header:** Specify the request header. You can add multiple request headers for same web service call; once you click on the "Add header element", you can specify the header request Key and Value.
- **Content Type:** Specify the content type.
- **Request Body:** Specify the request body.
- **Request Timeout:** Specify the request timeout in seconds, leaving it blank will set the request default value.
- **Format Data:** Specify the Format of Data to be retrieved, JSON or XML. This option will be invisible if SOAP is selected as SOAP always returns data in XML format.
- **Node Name:** Specify the Node Name of the selected Format Data (JSON or XML).
- **Field Name:** Specify the Field Name that you want to retrieve data in the form based on it.
- **Query Type:** Defines the control type you would like to represent the SQL data within, the types are: Searchable, Checkbox List, Dropdown List or Radio Button.

  Note: The system will automatically create a new rule when the selected query type is Checkbox List, Dropdown List or Radio Button.
- **Show All Data When Search Box is Empty:** If checked, the applicable data will be retrieved automatically when open the search dialog in the run-mode without needs to specify the value in the search box (text-search criteria).
- **Run Button:** You can execute and test the specified Settings by clicking on this icon. The data will be retrieved on the result panel.

See also

Reference
clear() function
getWebConnectorData() function
generateWebConnectorData() function
executeWebQuery() function
getWebServerRelativeURL() function
getValue() function
setFocus() function
addAttribute() function
getAttributeValue() function
modifyAttributeValue() function

10.4.8 LDAP Connector Control

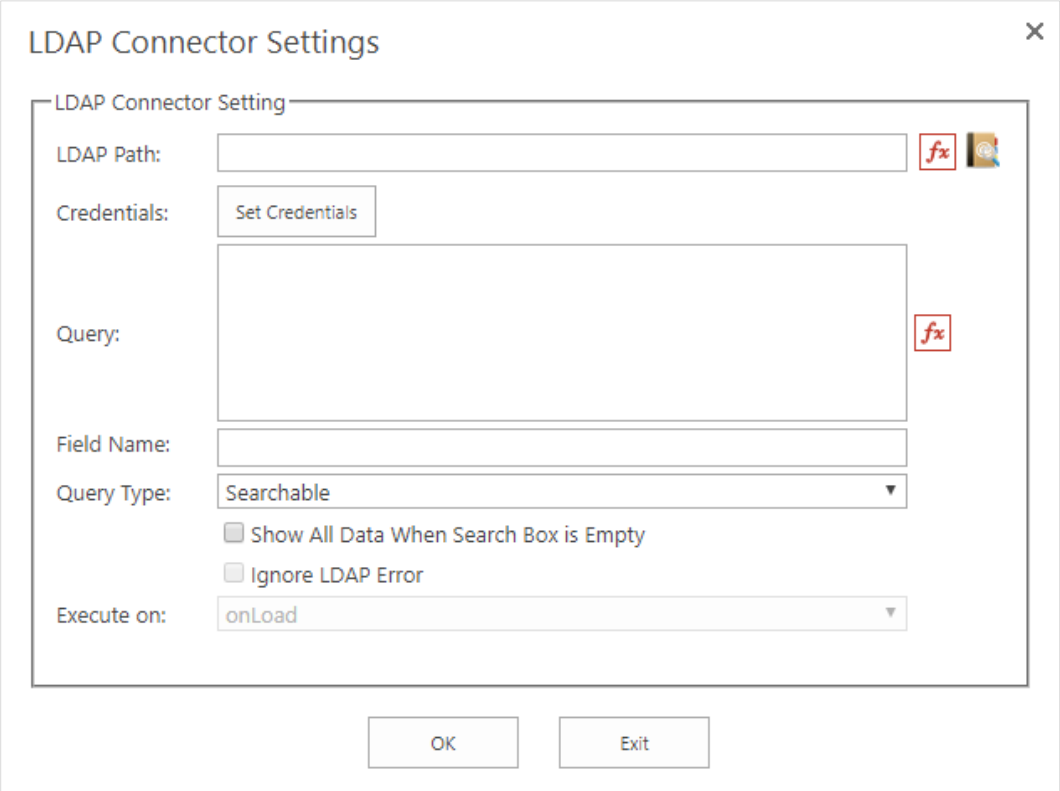
Note: This topic applies to SPARK Reports Builder Enterprise Edition only.

The **LDAP Connector** control allows users to read from a LDAP data source such as Microsoft Active Directory. You can consume the retrieved fields' values of the LDAP query using special functions.

Control Properties:

- **Control Type:** The type of the selected control.
- **ID:** The Identity of the control. The ID is a unique value and no duplicate IDs allowed in the form.
- **Tab Index:** Attribute specifies the tab order of an element (when the "tab" button is used for navigating).
- **Tooltip:** Appears on the running mode when the user moves the mouse pointer over the control.
- **CSS Classes:** The HTML class attribute makes it possible to define equal styles for elements with the same class name.
- **Style**: The HTML CSS code. The user can edit this code in the editor and change the control style in the designer and art runtime.
- **Control Settings:** For more details, refer to LDAP Connector Settings.
- **Dialog Width:** Specifies the width of the control's search/select dialog.
- **Dialog Height:** Specifies the height of the control's search/select dialog.
- **Header BG Color:** Specifies the color of the header's background in the control's search dialog.
- **Dialog Border Color:** Specifies the color of the border in the control's search dialog.
- **Dialog Border Size:** Specifies the size of the border in the control's search dialog.
- **Dialog Border Style:** Specifies the style of the border in the control's search dialog.
- **Required:** Specifies if the control will be required to be filled or not when the form loads. This property will not be active at runtime when both **Required** and **Hidden** properties are set to ON.
- **Required Border:** This property is visible only when the **Required** property is switched ON. This property allows the designer to choose if the red border would appear on the "required" control or not. If not (Required Border is switched OFF), the designer can keep the default border or change it as desired.
- **Disabled:** Specifies if the control will be disabled or not when the form loads.
- **ReadOnly:** Specifies if the control will be read only or not when the form loads.
- **Hidden:** Specifies if the control will be visible or not when the form loads.

LDAP Connector Setting:



- **LDAP Path:** Specify the LDAP path. You can specify this field manually or by clicking on (*fx)* button or by clicking on LDAP explorer button.
- **Credentials:** Click on "Set Credentials" button to specify the username/password to connect to the specified LDAP.
- **Query:** Specify the LDAP query to run. You can add dynamic attributes to the query by clicking on (*fx)* button.
- **Field Name:** Specify the field name that you want to retrieve data in the form based on it. For example, if the specified "Query Type" is a dropdown list, then the retrieved values in the dropdown list will be the values of this field.
- **Query Type:** Defines the control type you would like to represent the SQL data within, the types are: Searchable, Checkbox List, Dropdown List or Radio Button.

  Note: The system will automatically create a new rule when the selected query type is Checkbox List, Dropdown List or Radio Button.
- **Show All Data When Search Box is Empty:** If checked, the applicable data will be retrieved automatically when open the search dialog in the run-mode without needs to specify the value in the search box (text-search criteria).
- **Ignore LDAP Error:** Ignore any LDAP error at runtime.
- **Execute on:** Defines the way this control executes the LDAP through. The execution trigger could be onLoad or onChange. This attribute is only enabled if the selected query type is Checkbox List, Dropdown List or Radio Button.

See also

Reference
clear() function
getValue() function
executeLDAPQuery() function
generateLDAPConnectorData() function
getLDAPConnectorData() function
setFocus() function
addAttribute() function

getAttributeValue() function
modifyAttributeValue() function

10.4.9 Managed Metadata Control

Note: This topic applies to SPARK Reports Builder Enterprise Edition only.

The **Managed Metadata** control retrieves the managed terms defined in your SharePoint environment. For more information about managed metadata, please refer to the following link: https://docs.microsoft.com/en-us/SharePoint/governance/managed-metadata-planning.

Control Properties:

- **Control Type:** The type of the selected control.
- **ID:** The Identity of the control. The ID is a unique value and no duplicate IDs allowed in the form.
- **Tab Index:** Attribute specifies the tab order of an element (when the "tab" button is used for navigating).
- **Tooltip:** Appears on the running mode when the user moves the mouse pointer over the control.
- **CSS Classes:** The HTML class attribute makes it possible to define equal styles for elements with the same class name.
- **Style:** The HTML CSS code. The user can edit this code in the editor and change the control style in the designer and art runtime.
- **Multiple Selection:** This property is visible if the data source property's value was "Unmapped". Switch On if you want this control to perform a multiple selections.
- **Display Entire Path:** This property is visible if the data source property's value was "Unmapped". Switch On if you want have the full path of the managed metadata value.
- **Allow Fill-in:** This property is visible if the data source property's value was "Unmapped". Switch On if you want allow filling the control value manually.
- **Term Set:** Specify the managed metadata term set source form the terms selector dialog.
- **Required:** Specifies if the control will be required to be filled or not when the form loads. This property will not be active at runtime when both **Required** and **Hidden** properties are set to ON.
- **Required Border:** This property is visible only when the **Required** property is switched ON. This property allows the designer to choose if the red border would appear on the "required" control or not. If not (Required Border is switched OFF), the designer can keep the default border or change it as desired.
- **Disabled:** Specifies if the control will be disabled or not when the form loads.
- **Hidden:** Specifies if the control will be visible or not when the form loads.

See also

Reference
getManagedMetadataValue() function
getValue() function
setValue() function
clear() function
setFocus() function
addAttribute() function
getAttributeValue() function
modifyAttributeValue() function
show() function

11 Functions

Important: *All SPARK functions' names are case sensitive. For example, setValue() is the correct function name while SetValue() or setvalue() will not be recognized as an internal function.*

11.1 Financial Functions

Function Name	**avg()**
Format	avg(value0, value1, ..., N) Or avg([])
Description	For calculating the average of values.
Arguments	**value0-N:** The values that you need to calculate their average, any value could be dynamic from controls [at least one argument needed]. **array[]:** The array that contains the values, any of these values could be dynamic from control [at least one argument needed].
Argus Type	TextBox, Currency, TextArea, Rich Text Editor, DropDownList, Lookup, Label, Query Columns
Return	Number
Example	avg(4, 3, 2, 5) avg([TextBox1, 5, 1, 3])
Function Name	**sum()**
Format	sum(value0, value1, ..., N) Or sum([])
Description	For calculating the sum of values.
Arguments	**value0-N:** The values that you need to sum, any value could be dynamic from controls [at least one argument needed]. **array[]:** The array that contains the values, any value could be dynamic from controls [at least one argument needed].
Argus Type	TextBox, Currency, TextArea, Rich Text Editor, DropDownList, Lookup, Label, Query Columns
Return	Number
Example	sum(4, 5, 5, 5) sum([TextBox1, 5, 5, 5])
Function Name	**math()**
Format	math(Expression)
Description	To be Used for processing math expression.
Arguments	**Experssion:** The math expression that you need to process.
Argus Type	TextBox, Currency, TextArea, Rich Text Editor, DropDownList, Lookup, Label, Query Columns
Return	Number
Example	math(TextBox1+2)
Function Name	**max()**
Format	max(value0, value1, ..., N) Or max([])
Description	For getting the max value of a group of values.
Arguments	**value0-N:** The values to get their maximum one, any value could be dynamic from controls [at least one argument needed]. **[]:** The array that contains the values, any value could be dynamic from controls [at least one argument needed].
Argus Type	TextBox, Currency, TextArea, Rich Text Editor, DropDownList, Lookup, Label, Query Columns
Return	Number
Example	max(4, 3, 2, 5) max([TextBox1, 5, 1, 2])
Function Name	**min()**
Format	min(value0, value1, ..., N) Or min([])

Description	To get the min value of a group of values.
Arguments	**value0-N:** The values to get their minimum, any value could be dynamic from controls [at least one argument needed]. **[]:** The array that contains the values, any value could be dynamic from controls [at least one argument needed].
Argus Type	TextBox, Currency, TextArea, Rich Text Editor, DropDownList, Lookup, Label, Query Columns
Return	Number
Example	min(4, 3, 2, 5) min([TextBox1, 5, 1, 2])
Function Name	**executeFuncForEntireColData()**
Format	executeFuncForEntireColData(FunctionName)
Description	Execute the aggregates on all column value.
Arguments	**FunctionName:** Aggregates Function.
Argus Type	none
Return	Number
Example	executeFuncForEntireColData(sum(Col1))

11.2 Logical and Conditional Functions

Function Name	**and()**
Format	and(Bool1, Bool2)
Description	To execute AND operator
Arguments	**Bool1:** The first Boolean that you need to compare it, could be dynamic from controls. **Bool2:** The second Boolean that you need to compare it, could be dynamic from controls.
Argus Type	TextBox, Currency, TextArea, Rich Text Editor, Date, Time, DropDownList, Lookup, Label, HTMLText, Query Columns
Return	Boolean
Example	and(true, false)
Function Name	**or()**
Format	or(Bool1, Bool2)
Description	To execute OR operator
Arguments	**Bool1:** The first Boolean that you need to compare, it could be dynamic from controls. **Bool2:** The second Boolean that you need to compare, it could be dynamic from controls.
Argus Type	TextBox, Currency, TextArea, Rich Text Editor, Date, Time, DropDownList, Lookup, Label, HTMLText, Query Columns
Return	Boolean
Example	or(true, false)
Function Name	**not()**
Format	not(bool1)
Description	Returns false if its single operand can be converted to true; otherwise, returns true.
Arguments	**bool1:** The Boolean that you need to check it, could be dynamic from controls.
Argus Type	TextBox, Currency, TextArea, Rich Text Editor, Date, Time, DropDownList, Lookup, Label, HTMLText, Query Columns
Return	Boolean
Example	not(true)
Function Name	**isEqual()**
Format	isEqual(Value1, Value2)
Description	To determine if two values are equal.
Arguments	**Value1:** The first value you need to compare, it could be dynamic

	from controls. **Value2:** The second value you need to compare, it could be dynamic from controls.
Argus Type	TextBox, Currency, TextArea, Rich Text Editor, Date, Time, DropDownList, Lookup, Label, HTMLText, Query Columns
Return	Boolean
Example	isEqual(TextBox1, SPARKnit)
Function Name	**isNotEqual()**
Format	isNotEqual(Value1, Value2)
Description	To determine if two values are not equal.
Arguments	**Value1:** The first value you need to compare, it could be dynamic from controls. **Value2:** The second value you need to compare, it could be dynamic from controls.
Argus Type	TextBox, Currency, TextArea, Rich Text Editor, Date, Time, DropDownList, Lookup, Label, HTMLText, Query Columns
Return	Boolean
Example	isNotEqual(TextBox1, 'SPARKnit')
Function Name	**isGreaterThan()**
Format	isGreaterThan(Value1, Value2)
Description	To determine if first value is greater than second value.
Arguments	**Value1:** The first value you need to compare, it could be dynamic from controls. **Value2:** The second value you need to compare, it could be dynamic from controls.
Argus Type	TextBox, Currency, TextArea, Rich Text Editor, Date, Time, DropDownList, Lookup, Label, HTMLText, Query Columns
Return	Boolean
Example	isGreaterThan(TextBox1, 6)
Function Name	**isGreaterEqualThan()**
Format	isGreaterEqualThan(Value1, Value2)
Description	To determine if first value is greater than or equal to second value.
Arguments	**Value1:** The first value you need to compare, it could be dynamic from controls. **Value2:** The second value you need to compare, it could be dynamic from controls.
Argus Type	TextBox, Currency, TextArea, Rich Text Editor, Date, Time, DropDownList, Lookup, Label, HTMLText, Query Columns
Return	Boolean
Example	isGreaterThan(TextBox1, 6)
Function Name	**isNumber()**
Format	isNumber(Value)
Description	To determine if the value is number or not.
Arguments	**Value:** The value you need to check, it could be dynamic from controls.
Argus Type	TextBox, Currency, TextArea, Rich Text Editor, Date, Time, DropDownList, Lookup, Label, HTMLText, Query Columns
Return	Boolean
Example	isNumber(TextBox1)
Function Name	**isSmallerThan()**
Format	isSmallerThan(Value1, Value2)
Description	To determine if first value is smaller than second value.
Arguments	**Value1:** The first value you need to compare, it could be dynamic from controls. **Value2:** The second value you need to compare, it could be dynamic value from controls.
Argus Type	TextBox, Currency, TextArea, Rich Text Editor, Date, Time,

	DropDownList, Lookup, Label, HTMLText, Query Columns
Return	Boolean
Example	isSmallerThan(TextBox1, 6)
Function Name	**isSmallerEqualThan()**
Format	isSmallerEqualThan(Value1, Value2)
Description	To determine if first value is smaller than or equal to second value.
Arguments	**Value1:** The first value you need to compare it, could be dynamic from controls. **Value2:** The second value you need to compare it, it could be dynamic value from controls.
Argus Type	TextBox, Currency, TextArea, Rich Text Editor, Date, Time, DropDownList, Lookup, Label, HTMLText, Query Columns
Return	Boolean
Example	isSmallerEqualThan(TextBox1, 6)

11.3 Controls Functions

11.3.1 General Controls

11.3.1.1 Hyperlink Functions

Function Name	**setHyperLink()**
Format	setHyperLink(ControlId, Title, Link)
Description	To set the hyperlink control href and its title.
Arguments	**ControlId:** The control id you need to set the href and title values for it. **Title:** The hyperlink title you need to set, it could be dynamic value from controls. **Link:** The hyperlink href value you need to set (URL), could be dynamic from controls.
Argus Type	Hyperlink, TextBox, Currency, TextArea, Rich Text Editor, DropDownList, Lookup, Label, HTMLText.
Return	None
Example	setHyperLink(HyperLink1, 'SPARKnit', 'http://www.sparknit.com/')

11.3.1.2 Image Functions

Function Name	**setImage()**
Format	setImage(ControlId, ImgSrc)
Description	To set the image control source value.
Arguments	**ControlId:** The image control id you need to set the source value for it. **ImgSrc:** The Image source value you need to set (URL), could be dynamic from controls.
Argus Type	Image, TextBox, TextArea, Rich Text Editor, DropDownList, Lookup, Label, HTMLText.
Return	None
Example	setImage(Image1, 'images/EmptyPicture1.png')
Function Name	**imageSelector()**
Format	imageSelector(ControlId)
Description	This function will open the "SharePoint image selector popup dialog" to select an image from and set its URL to the Image URL property of the Image control. The system returns only images in the "image selector popup dialog".
Arguments	**ControlId:** The control id and should be Image control only. The system will set automatically the URL of the selected image to the Image URL property of the Image control.

Argus Type	Image
Return	None
Example	imageSelector(Image1)
Function Name	**linkSelector()**
Format	linkSelector(ControlId)
Description	This function will open the "SharePoint image selector popup dialog" to select an image or any file type from and set its URL to the control value.
Arguments	**ControlId:** The control id that you want to set the control value to, for example, if the control is HyperLink, the value of the selected URL will be set to the control value of the HyperLink, if the control is a TextBox control, then the value of the selected URL will be set to the control value of the TextBox and so on.
Argus Type	HyperLink, TextBox, TextArea, Label, Rich Text Editor
Return	None
Example	linkSelector(HyperLink)

See also

Reference

Other resources

Video: https://youtu.be/KNcuCTnLl1c.This video will show you how you can add SQL connector control in your form and integrate it with external databases directly without the need to use SharePoint external connectivity services, and how to configure the control to retrieve data from external databases and map the retrieved data to multiple controls in your form at runtime.
You will see how to create a dynamic SQL Query and how to combine your query with dynamic objects such as form's controls and form's variables and even with static values instantly in order to integrate the data selected from this query with the form's controls and list columns.
This video will show you how to work with the SQL Connector control advanced features, by binding the query data into 4 types of controls, the SQL connector control will render its GUI according to your selection and configuration, and the designer can choose to render the control to be a searchable dialog, drop-down list, checkbox list or even a radio buttons list control.

11.3.1.3 PageViewer Functions

Function Name	**setPageViewer()**
Format	setPageViewer(ControlId, src)
Description	To set the PageViewer control source value.
Arguments	**ControlId:** The PageViewer control id you need to set the source value for it. **src:** The PageViewer source value you need to set (URL), could be dynamic from controls.
Argus Type	PageViewer, TextBox, Currency, TextArea, Rich Text Editor, DropDownList, Lookup, Label, HTMLText.
Return	None
Example	setPageViewer(PageViewer1, 'http://www.sparknit.com/')

11.3.2 Input Controls

11.3.2.1 CheckBoxList Functions

Function Name	**addCheckBoxListItem()**
Format	addCheckBoxListItem(ControlId, item)
Description	To add a new checkbox in the CheckBoxList's control.
Arguments	**ControlId:** The Checkboxlist control id that you need to add the new checkbox inside.

	Item: string contains the name of the new checkbox, could be dynamic from controls.
Argus Type	Checkboxlist, TextBox, Currency, TextArea, Rich Text Editor, Date, Time, DropDownList, Lookup, Label, HTMLText.
Return	None
Example	addCheckBoxListItem(CheckBoxList1, 'CheckBoxName1')
Function Name	**removeCheckBoxListItemByIndex()**
Format	removeCheckBoxListItemByIndex(ControlId, item)
Description	To remove a checkbox from the checkboxlist control by using the index.
Arguments	**ControlId:** The Checkboxlist control id that you need to remove the checkbox from within. **item:** Index number of the checkbox you need to remove, could be dynamic from controls.
Argus Type	Checkboxlist, TextBox, Currency, TextArea, Rich Text Editor, Date, Time, DropDownList, Lookup, Label, HTMLText.
Return	None
Example	removeCheckBoxListItemByIndex(CheckBoxList1, 1)
Function Name	**removeCheckBoxListItemByName()**
Format	removeCheckBoxListItemByName(ControlId, item)
Description	To remove a checkbox from the checkboxlist control by using the name.
Arguments	**ControlId:** The Checkboxlist control id that you need to remove the checkbox from within. **item:** String contains the checkbox name you need to remove, could be dynamic from controls.
Argus Type	Checkboxlist, TextBox, Currency, TextArea, Rich Text Editor, Date, Time, DropDownList, Lookup, Label, HTMLText.
Return	None
Example	removeCheckBoxListItemByName(CheckBoxList1, 'CheckBoxItem1')
Function Name	**getCheckBoxListItems()**
Format	getCheckBoxlistItems(ControlId)
Description	Retrieves a string that represents the checked checkboxes separated by commas.
Arguments	**ControlId:** The control id you need to retrieve its checked values.
Argus Type	Checkboxlist
Return	String
Example	getCheckBoxlistItems(CheckBoxList1)
Function Name	**selectAtLeast()**
Format	selectAtLeast(CheckBoxListId, NumberOfItems)
Description	To determine how many items must be selected at least.
Arguments	**CheckBoxListId:** The control id that will be checked. **NumberOfItems:** The min number of the selected items.
Argus Type	CheckBoxList
Return	Boolean
Example	selectAtLeast(checkboxlist10, 3)
Function Name	**selectAtMost()**
Format	selectAtMost(CheckBoxListId, NumberOfItems)
Description	To determine how many items must be selected at most.
Arguments	**CheckBoxListId:** The control id that will be checked. **NumberOfItems:** The max number of the selected items.
Argus Type	CheckBoxList
Return	Boolean
Example	selectAtMost(checkboxlist10, 3)
Function Name	**selectExactly()**
Format	selectExactly(CheckBoxListId, NumberOfItems)

Description	To determine exactly how many items must be selected at most.
Arguments	**CheckBoxListId:** The control id that will be checked. **NumberOfItems:** The exact number of the selected items.
Argus Type	CheckBoxList
Return	Boolean
Example	selectExactly(checkboxlist10, 2)
Function Name	**setCheckBoxListItemByIndex()**
Format	setCheckBoxListItemByIndex(ControlId, flag, items)
Description	To check or uncheck the checkboxes inside the CheckBoxList.
Arguments	**ControlId:** The CheckBoxList control id you need to check or uncheck its checkboxes. **flag:** Boolean. **items:** Index numbers of checkboxes separated by commas.
Argus Type	CheckBoxList
Return	None
Example	setCheckBoxListItemByIndex(CheckBoxList1, true, '1,2,3')
Function Name	**setCheckBoxListItemByName()**
Format	setCheckBoxListItemByName(ControlId, flag, items)
Description	To check or uncheck the checkboxes inside the CheckBoxList.
Arguments	**ControlId:** The CheckBoxList control id you need to check or uncheck its checkboxes. **flag:** Boolean. **items:** string contains the name of checkboxes separated by commas.
Argus Type	CheckBoxList
Return	None
Example	setCheckBoxListItemByName(CheckBoxList1, true, 'CheckBoxItem1, CheckBoxItem2, CheckBoxItem3')

11.3.2.2 DropDownList Functions

Function Name	**addDropDownItem()**
Format	addDropDownItem(ControlId, item)
Description	To add a new item inside the DropDownList control.
Arguments	**ControlId:** The DropDownList control id that you need to add the new item inside. **item:** string contains the name of the new item, could be dynamic from controls.
Argus Type	TextBox, Currency, TextArea, Rich Text Editor, Date, Time, DropDownList, Lookup, Label, HTMLText.
Return	None
Example	addDropDownItem(ListBox1, 'item1')
Function Name	**removeDropDownItem()**
Format	removeDropDownItem(ControlId, 'item')
Description	To remove an item from within the DropDownList control.
Arguments	**ControlId:** The DropDownList control id that you need to remove the item from it. **Item:** String contains the item name you need to remove, could be dynamic from controls.
Argus Type	TextBox, Currency, TextArea, Rich Text Editor, Date, Time, DropDownList, Lookup, Label, HTMLText.
Return	None
Example	removeDropDownItem(ListBox1, 'item0')

11.3.2.3 CheckBox Functions

Function Name	**getCheckBox()**

Format	getCheckBox(ControlId)
Description	Returns the status of the CheckBox (checked/unchecked).
Arguments	**ControlId:** The control id you need to returns its status.
Argus Type	CheckBox
Return	Boolean
Example	getCheckBox(CheckBox1)
Function Name	**setCheckBox()**
Format	setCheckBox(ControlId, flag)
Description	To check or uncheck the CheckBox control.
Arguments	**ControlId:** The checkbox control id you need to check or uncheck. **Flag:** Boolean.
Argus Type	CheckBox
Return	None
Example	setCheckBox(CheckBox1, true)
Function Name	**checked()**
Format	checked(CheckboxId,'Flag','item1','item2',.....,'N').
Description	To checked/unckeced checkbox control or multi checkbox's inside checkboxlist.
Arguments	**ControlId:** The control id to be checked. **Flag:** Set true that's mean checked or set false that's mean unchecked. **item0_N**: The items label that you need to checked/unckeced inside the ckeckboxlist.[optional].
Argus Type	Checkbox, Checkboxlist.
Return	None
Example	checked(checkboxlist1,'true','item0','item1')

11.3.2.4 Radio Button Functions

Function Name	**setRadioButton()**
Format	setRadioButton(ControlId, ItemLabel)
Description	To set value to RadioButton control.
Arguments	**ControlId:** The radio control id you need to set it. **ItemLabel:** The radio's item that want to check.
Argus Type	RadioButton
Return	None
Example	setRadioButton(RadioButton1, 'ItemLabel0')
Function Name	**getRadioButton()**
Format	getRadioButton(ControlId)
Description	Returns the value of the RadioButton control.
Arguments	**ControlId:** The radio id you need to return its checked item.
Argus Type	RadioButton
Return	The Label of the checked item
Example	getRadioButton(RadioButton1)
Function Name	**addRadioButtonItem()**
Format	addRadioButtonItem(ControlId, item)
Description	To add a new radio value inside the RadioButton control.
Arguments	**ControlId:** The RadioButton control id that you need to add the new radio inside. **Item:** String represents the value of the new radio, could be dynamic from controls.
Argus Type	Checkboxlist, TextBox, Currency, TextArea, RichTextBox, Date, Time, DropDownList, Lookup, Label, HTMLText.
Return	None
Example	addRadioButtonItem(RadioButton1,'item1')

11.3.2.5 Date & Time Functions

Function Name	**calculateDays()**
Format	calculateDays(FirstDateID, SecondDateId)
Description	To calculate the difference between two dates.
Arguments	**FirstDateID:** The control id of first date. **SecondDateId:** The control id of second date.
Argus Type	Date
Return	Integer
Example	calculateDays(dateC1, dateC2)
Function Name	**currentDate()**
Format	currentDate()
Description	Retrieve the current date
Arguments	None
Argus Type	None
Return	The current date
Example	currentDate()
Function Name	**extractDateValue()**
Format	extractdatevalue(DateValue, DatePart)
Description	To extract a date part from a specific Date control or DateTime control.
Arguments	**DateValue:** Specify the date value to extract the date part from. It could be dynamic from control. **DatePart:** Specify the date part to be extracted. The date part can be one of the following values: Year, Month, MonthName, MonthFullName, Day, DayName, DayFullName, DayWeek, DayYear, WeekNumber, NumberOfDays, ISODate
Argus Type	TextBox, TextArea, RichBox, Date, DateTime, DropDownList, Lookup, Label, HTMLText.
Return	The specified date part.
Example	extractdatevalue('11/25/2018', 'MonthName')
Function Name	**addDateTime()**
Format	addDateTime(DateTime, DateTimePart, Number)
Description	To add or subtract value from a specific date, datetime or time.
Arguments	**DateTime:** The value of date, datetime or time. It could be dynamic from control. **DateTimePart:** take one of the following values: Years, Months, Days, Hours and Minutes. **Number:** The number that you want to add or subtract the DateTimePart to/from the given DateTime. Negative or positive integer value.
Argus Type	TextBox, TextArea, RichBox, Date, Time, DateTime, DropDownList, Lookup, Label, HTMLText.
Return	New date
Example	addDateTime(Date1,'Year',2) will add two years to Date1.
Function Name	**differenceDateTime()**
Format	differenceDateTime(StartDateTime, EndDateTime, Format)
Description	Returns the difference between two dates or two datetimes in a specific format.
Arguments	**StartDateTime:** Specify the first date or datetime. **EndDateTime:** Specify the second date or datetime. **Format:** Format takes one of the following values: Years, Months, Days, Hours and Minutes.
Argus Type	TextBox, TextArea, RichBox, Date, DateTime, DropDownList, Lookup, Label, HTMLText.
Return	Number
Example	differenceDateTime('10/10/2018','10/12/2018','Years') will return 2.
Function Name	**calculateDaysExcludedDays()**

Format	calculateDaysExcludedDays(StartDate, EndDate, ExcludedDays, WeekendDays)
Description	To calculate number of days between two dates or datetimes excluding specific dates and/or weekends.
Arguments	**StartDate:** Specify the first date. **EndDate:** Specify the second date. **ExcludedDays:** Optionally specify dates, as an array, to exclude from the calculation. **WeekendDays:** Optionally specifies weekend day names, as an array, to be excluded from the calculation. Note: If a day, in ExcludedDays, came in the specified weekend days, this day will be excluded from the calculation.
Argus Type	TextBox, TextArea, RichBox, Date, DropDownList, Lookup, Label, HTMLText.
Return	Number
Example	calculateDaysExcludedDays('10/10/2018', '10/15/2018', ['10/10/2018', '10/15/2018'], ['Sunday','Saturday'])
Function Name	**weekendBetweenTwoDates()**
Format	weekendBetweenTwoDates(StartDate, EndDate, WeekendDays)
Description	To return an array of weekend dates between two dates or datetimes.
Arguments	**StartDate:** Specify the first date (date from). **EndDate:** Specify the end date (date to). **WeekendDays:** Specify the weekend day names as an array.
Argus Type	TextBox, TextArea, RichBox, Date, DropDownList, Lookup, Label, HTMLText.
Return	Array
Example	weekendBetweenTwoDates('10/10/2018', '10/15/2018', ['Saturday', 'Sunday']) → will return an array of '10/13/2018', '10/14/2018', '10/20/2018', '10/21/2018'
Function Name	**compareDate()**
Format	compareDate(StartDate, EndDate, Operator)
Description	To compare between two dates or two datetimes.
Arguments	**StartDate:** Specify the first date and/or datetime to compare with. The control value could be dynamic. **EndDate:** Specify the second date and/or datetime to compare with. The control value could be dynamic. **Operator:** Specify the operator that will use to compare between the specified dates and/or datetimes as follows: eq: equal neq: not equal gt: greater than gte: greater than or equal lt: less than lte: less than or equal
Argus Type	TextBox, TextArea, RichBox, Date, DateTime, DropDownList, Lookup, Label, HTMLText.
Return	Boolean
Example	compareDate('10/10/2018', '10/15/2018','gt'). → returns "false". compareDate(Date1, Date2, 'lte'). compareDate(Date1, DateTime1, 'gte').

11.3.2.6 Currency Functions

Function Name	**getCurrencyValue()**
Format	getCurrencyValue(ControlId, Numeric_Flag)
Description	To get the currency control value as a number or as a currency format based on the specified flag.
Arguments	**ControlId:** The currency control id that you need to get its value. **Numeric_Flag:** Used to detect if you need to get its numeric value (true) or currency format text.
Argus Type	Currency
Return	String or Number
Example	getCurrencyValue(TextBox1, true) getCurrencyValue(TextBox1, false)

11.3.2.7 RichText Editor Functions

Function Name	**appendRichTextEditor()**
Format	appendRichTextEditor(ControlId, Text)
Description	To append a string value into the RichTextEditor or HTMLText controls source value
Arguments	**ControlId:** The RichTextEditor or HTMLText control id. **Text:** The text that want to append, could be dynamic from controls.
Argus Type	Image, TextBox, Currency, TextArea, RichBox, DropDownList, Lookup, Label, HTMLText.
Return	None
Example	appendRichTextEditor(RichTextEditor1,'Text to append')

11.3.3 Container Controls

11.3.3.1 Tab Functions

Function Name	**disableTab()**
Format	disableTab(TabID, TabName, flag)
Description	To disable a single tab in a tab control.
Arguments	**TabID:** Specify the Tab control ID. **TabName:** Specify the tab name that you want to disable. **Flag:** true ==> disable, false==>enable.
Argus Type	Tab
Return	None
Example	disableTab(Tab1,'TabName1',true)
Function Name	**hideTab()**
Format	hideTab(TabID, TabName, flag)
Description	To hide a single tab in a tab control.
Arguments	**TabID:** Specify the Tab control ID. **TabName:** Specify the tab name that you want to hide. **showTab:** Specify the tab that will be show after the TabName is hidden. **Flag:** true ==> hide, false==>show.
Argus Type	Tab
Return	None
Example	hideTab(Tab1,'TabName1', 'TabName3', true)
Function Name	**setActiveTab()**
Format	setActiveTab(TabID, TabName)
Description	To activate a specific Tab in a Tab control.
Arguments	**TabID:** Specify the Tab control ID. **TabName:** Specify the tab name that you want to set active.
Argus Type	Tab
Return	None

Example	setActiveTab(Tab1,'TabName1')
Function Name	**tabHeaderCaptionColor ()**
Format	tabHeaderCaptionColor(TabID, TabName, color)
Description	To set color for a Tab header caption.
Arguments	**TabID:** Specify the Tab control ID. **TabName:** Specify the tab name that you want to change its header's color. Color: The color value you want to set. The color value must be Hexadecimal or color name.
Argus Type	Tab
Return	None
Example	tabHeaderCaptionColor(Tab1,'TabName1','red')

11.3.4 Integration Controls

11.3.4.1 SQL Connector Functions

Function Name	**getSQLData()**
Format	getSQLData(ControlId, RetrievedField)
Description	To get a SQL field's value from SQL Connector data.
Arguments	**ControlId:** The SQL connector id that you want to get its retrieved fields values. **RetrievedField:** The name of the field that want to retrieve its value.
Argus Type	SQL Connector
Return	String or array
Example	getSQLData(SQLConnector1, 'EmployeeName')
Function Name	**generateSQLData()**
Format	generateSQLData(ControlId, RetrievedFields, ControlsToFill)
Description	To get a SQL field's values and assign to any form controls.
Arguments	**ControlId:** The SQL connector id that you want to get its retrieved fields values. **RetrievedField:** Array contains the name of SQL fields. **ControlsToFill:** Array contains the id of controls, mapping with RetrievedFields array.
Argus Type	SQL Connector
Return	None
Example	generateSQLData(SQLConnector1, ['EmployeeName', 'EmployeeNo'], ['NameId', NumberId'])
Function Name	**getSQLProcedureData()**
Format	getSQLProcedureData(ControlId)
Description	Retrieves the data generated by executing the SQL procedure as specified in SQL Connector Control Settings property.
Arguments	**ControlId:** The SQL Connector control id which will execute the SQL procedure.
Argus Type	SQL Connector
Return	JSON
Example	getSQLProcedureData(SQLConnector2)
Function Name	**executeSQLQuery()**
Format	executeSQLQuery(ControlId,functionName)
Description	Execute SQL query as specified in the Control Settings properties of the SQL Connector control.
Arguments	**ControlID:** The SQL Connector. **functionName:** (Optional) This function will execute after executing what is specified when configuring the SQL Connector

	control.
Argus Type	SQL Connector
Return	None
Example	Example #1:executeSQLQuery(SQLConnector2) Example #2: function doCallBack(){}; executeSQLQuery(SQLConnector2, doCallBack)

11.3.4.2 ECT Functions

Function Name	**getECTData()**
Format	getECTData(ControlId, RetrievedField)
Description	To get an external dialog field's value from External Data Dialog.
Arguments	**ControlId:** The External Dialog id that you want to get its retrieved fields values. **RetrievedField:** The name of a field that want to retrieve its value.
Argus Type	External Data Dialog
Return	String or array
Example	getECTData(ExteranlDialog1, 'EmployeeName')
Function Name	**generateECTData()**
Format	generateECTData(ControlId, RetrievedFields, ControlsToFill)
Description	To get an external dialog field's values and populate it to any form controls
Arguments	**ControlId:** The External Data Dialog id that you want to get its retrieved fields values **RetrievedField:** Array contains the name of External Data Dialog fields **ControlsToFill:** Array contains the id of controls mapped with RetrievedFields array
Argus Type	External Data Dialog
Return	None
Example	generateECTData(ExternalDialog1, ['EmployeeName', 'EmployeeNo'], ['NameId', NumberId'])

11.3.4.3 Lookup Functions

Function Name	**getLookupSelectedText()**
Format	getLookupSelectedText(ControlId)
Description	To get the current or the selected value of the Lookup control.
Arguments	**ControlId:** The control id that you need to get its value.
Argus Type	Lookup
Return	String
Example	getLookupSelectedText(Lookup1)

11.3.4.4 Advanced Lookup Functions

Function Name	**getAdvancedLookupData()**
Format	getAdvancedLookupData(ControlId, RetrievedField)
Description	To get an advanced lookup field's value from Advanced Lookup data.
Arguments	**ControlId:** The Advanced Lookup id that you want to get its retrieved field value from. **RetrievedField:** The name of the field that you want to retrieve its value in.
Argus Type	Advanced Lookup
Return	String or array
Example	getAdvancedLookupData(AdvancedLookup1, 'EmployeeName')

Function Name	**generateAdvancedLookupData()**
Format	generateAdvancedLookupData(ControlId, RetrievedFields, ControlsToFill)
Description	To get an advanced Lookup field's values and assign them to specified form controls.
Arguments	**ControlId:** The Advanced Lookup control id that you want to get its retrieved fields values from. **RetrievedField:** Array contains the name of Advanced Lookup fields. **ControlsToFill:** Array contains the id of controls mapping with RetrievedFields array.
Argus Type	Advanced Lookup
Return	None
Example	generateAdvancedLookupData(AdvancedLookup1, ['EmployeeName', 'EmployeeNo'], ['NameId', NumberId'])

11.3.4.5 Web Connector Functions

Function Name	**getWebConnectorData()**
Format	getWebConnectorData(ControlId, RetrievedField)
Description	To get a web service field's value from Web Connector data.
Arguments	**ControlId:** The Web Connector id that you want to get its retrieved field value from. **RetrievedField:** The name of the field that you want to retrieve its value in.
Argus Type	Web Connector
Return	String or array
Example	getWebConnectorData(WebConnector1, 'EmployeeName')
Function Name	**generateWebConnectorData()**
Format	generateWebConnectorData(ControlId, RetrievedFields, ControlsToFill).
Description	To get a web service field's values and assign them to specified form controls.
Arguments	**ControlId:** The Web Connector id that you want to get its retrieved fields values from. **RetrievedField:** Array contains the name of web service fields. **ControlsToFill:** Array contains the id of controls, mapping with RetrievedFields array
Argus Type	Web Connector
Return	None
Example	generateWebConnectorData(WebConnector1, ['EmployeeName', 'EmployeeNo'], ['NameId', NumberId'])

11.3.4.6 XML Connector Functions

Function Name	**getXMLData()**
Format	getXMLData(ControlId, RetrievedField)
Description	To get a XML field's value from XML Connector data.
Arguments	**ControlId:** The XML connector id that you want to get its retrieved field value from. **RetrievedField:** The name of the field that you want to retrieve its value in.
Argus Type	XML Connector
Return	String or Array
Example	getXMLData(XMLConnector1, 'EmployeeName'
Function Name	**generateXMLData()**
Format	generateXMLData(ControlId, RetrievedFields, ControlsToFill)

Description	To get XML field's values and assign them to the specified form controls.
Arguments	**ControlId:** The XML connector id that you want to get its retrieved fields values from. **RetrievedField:** Array contains the name of XML fields. **ControlsToFill:** Array contains the id of controls, mapping with RetrievedFields array.
Argus Type	XML Connector
Return	None
Example	generateXMLData(XMLConnector1, ['EmployeeName', 'EmployeeNO'], ['NameId', NumberId'])

11.3.4.7 LDAP Connector Functions

Function Name	**executeLDAPQuery()**
Format	executeLDAPQuery(ControlID, functionName)
Description	Execute LDAP query as specified in the Control Settings properties of the LDAP Connector control.
Arguments	**ControlId:** The LDAP connector id. **functionName:** (Optional) This function will execute after executing what is specified when configuring the LDAP Connector control.
Argus Type	LDAP Connector
Return	None

Example	Example #1: executeLDAPQuery(LDAPConnector1);
	Example #2:
	function doCallBack(){
	/*Implement doCallBack that will call after executeLDAPQuery function
	};
	executeLDAPQuery(LDAPConnector1, doCallBack)

Function Name	**getLDAPConnectorData()**
Format	getLDAPConnectorData(ControlId,RetrievedField)
Description	To get a LDAP field's value from LDAP Connector data.
Arguments	**ControlId:** The LDAP connector id that you want to get its retrieved fields values from. **RetrievedField:** The name of the field that want to retrieve its value.
Argus Type	LDAP Connector
Return	String or array
Example	getLDAPConnectorData(LDAPConnector1, 'email')
Function Name	**generateLDAPConnectorData()**
Format	generateLDAPConnectorData(ControlId, RetrievedFields, ControlsToFill)
Description	To get a LDAP field's values and assign to any form controls.
Arguments	**ControlId:** The LDAP connector id that you want to get its retrieved fields values from. **RetrievedField:** Array contains the name of the LDAP fields. **ControlsToFill:** Array contains the id of controls, mapping with RetrievedFields array.
Argus Type	LDAP Connector
Return	None
Example	generateLDAPConnectorData(LDAPConnector1, ['Email'], ['NameId'])

11.3.4.8 Managed Metadata Functions

Function Name	**getManagedMetadataValue()**

Format	getManagedMetadataValue(ControlId, Part_Flag)
Description	To get the Managed Metadata term control value as a Name or GUID format based on the specified flag.
Arguments	**ControlId:** The Managed Metadata control id that you need to get its value. **Part_Flag:** Specify the form of value to be retrieved. 0: retrieves the term name, 1: retrieves the GUID of term, 2: retrieves both (Term Name and its GUID).
Argus Type	ManagedMetadata
Return	String
Example	getManagedMetadataValue(ManagedMetadata1, 0) getManagedMetadataValue(ManagedMetadata1, 1)

11.3.4.9 People Picker Functions

Function Name	**setPeoplePicker()**
Format	setPeoplePicker(ControlId, UsersNames)
Description	To set the People Picker control values.
Arguments	**ControlId:** The PeoplePicker control id you need to set the values for it. **UsersNames:** The users' names as presented in PeoplePicker control in SharePoint, this parameter can handle multiple values separated by commas. The value of this argument could be static or dynamic from form variable or form control.
Argus Type	People Picker
Return	None
Example	setPeoplePicker(PeoplePicker1, 'Jack')
Function Name	**getSiteCollectionUserID()**
Format	getSiteCollectionUserID(controlID)
Description	Retrieves the SharePoint site collection user Id specified in a People Picker control.
Arguments	**controlID:** The id of the PeoplePicker control that want to get its Id.
Argus Type	People Picker
Return	String or Array
Example	getSiteCollectionUserID(PeoplePicker1)
Function Name	**getPeoplePickerValue()**
Format	getPeoplePickerValue(controlID, valueType)
Description	To return the People Picker control value, the value could be account name, display text or email address depending on the given valueType. In case of multi values People Picker, the returned value will be array.
Arguments	**controlID:** The id of the control that you want to get its value. **valueType:** It is an integer index to specify which value to be returned. The valueType will be --> 1:Account Name, 2:DisplayText, 3:Email, 4:Key.
Argus Type	People Picker
Return	String or Array

Example	getUserProfileData(getPeoplePickerValue(PeoplePicker1, 1), ['AccountName', 'DisplayName', 'FirstName', 'CellPhone']);
	getPeoplePickerValue(PeoplePicker1, 2);
	getPeoplePickerValue(PeoplePicker1, 3)

11.4 Text Functions

Function Name	**concat()**
Format	concat(ControlId0, 'StringValue',.....,'N')
Description	To concatenate controls or string values into a single string.

Arguments	**ControlId0-N:** The ids of controls that you need to concatenate their values [at least one argument needed].
Argus Type	TextBox, Currency, TextArea, Date, Time, DropDownList, Lookup, Label, HTMLText, Query Columns.
Return	String
Example	concat(TextBox1, TextBox2, 'sparknit') will return AABBsparknit if the value of TextBox1 is AA and the value of TextBox2 is BB.
Function Name	**contain()**
Format	contain(Value, Text)
Description	To check if Value contain the given text.
Arguments	**Value:** The value you need to check it, could be dynamic from controls. **Text:** The text that you need to check if exists in Value, could be dynamic from controls.
Argus Type	TextBox, Currency, TextArea, Rich Text Editor, Date, Time, DropDownList, Lookup, Label, HTMLText, Query Columns.
Return	Boolean
Example	contain(TextBox1, 'SPARKnit')
Function Name	**notContain()**
Format	notContain(Value, Text)
Description	To check if the Value does not contain the given text.
Arguments	**Value:** The value you need to check it, could be dynamic from controls. **Text:** The text that you need to check if it is not exist in Value, could be dynamic from controls
Argus Type	TextBox, Currency, TextArea, Rich Text Editor, Date, Time, DropDownList, Lookup, Label, HTMLText, Query Columns.
Return	Boolean
Example	notContain(TextBox1, 'SPARKnit')
Function Name	**startWith()**
Format	startWith(Value, Prefix)
Description	To check if Value start with the given Prefix.
Arguments	**Value:** The value you need to check Prefix with, it could be dynamic from controls values. **Prefix:** The text that you need to check if Value start with, it could be dynamic from controls values.
Argus Type	TextBox, Currency, TextArea, Rich Text Editor Date, Time, DropDownList, Lookup, Label, HTMLText, Query Columns.
Return	Boolean
Example	startWith(TextBox1, 'SPARKnit')
Function Name	**notStartWith()**
Format	notStartWith(Value, Prefix)
Description	To check if the Value does not start with the given Prefix.
Arguments	**Value:** The value you need to check Prefix with, could be dynamic from controls. **Prefix:** The text that you need to check if Value does not start with it, could be dynamic from controls.
Argus Type	TextBox, Currency, TextArea, Rich Text Editor, Date, Time, DropDownList, Lookup, Label, HTMLText, Query Columns.
Return	Boolean
Example	notStartWith(TextBox1, 'SPARKnit')
Function Name	**endWith()**
Format	endWith(Value, Suffix)
Description	To check if the control Value ends with the given Suffix.
Arguments	**Value:** The value you need to check Suffix with, could be dynamic from controls. **Suffix:** The text you need to check if the Value end with it, could

	be dynamic from controls.
Argus Type	TextBox, Currency, TextArea, Rich Text Editor, Date, Time, DropDownList, Lookup, Label, HTMLText, Query Columns.
Return	Boolean
Example	endWith(TextBox1,'SPARKnit')
Function Name	**notEndWith()**
Format	notEndWith(Value, Suffix)
Description	To check if Value does not end with the given Suffix.
Arguments	**Value:** The value you need to check it, could be dynamic from controls. **Suffix:** The text that you need to check if Value not end with it, could be dynamic from controls.
Argus Type	TextBox, Currency, TextArea, Rich Text Editor, Date, Time, DropDownList, Lookup, Label, HTMLText, Query Columns.
Return	Boolean
Example	notEndWith(TextBox1, 'SPARKnit')
Function Name	**isEmpty()**
Format	isEmpty(Value)
Description	To determine if the value is empty or not.
Arguments	**Value:** The value you need to check, could be dynamic from controls.
Argus Type	TextBox, Currency, TextArea, Rich Text Editor, DropDownList, Lookup, Date, Time, Label, HTMLText, Query Columns.
Return	Boolean
Example	isEmpty(TextBox1)
Function Name	**isNotEmpty()**
Format	isNotEmpty(Value)
Description	To determine if the value is not empty.
Arguments	**Value:** The value you need to check, could be dynamic from controls.
Argus Type	TextBox, Currency, TextArea, Rich Text Editor, DropDownList, Lookup, Date, Time, Label, HTMLText, Query Columns.
Return	Boolean
Example	isNotEmpty(TextBox1)
Function Name	**replace()**
Format	replace(String, oldValue, newValue)
Description	To replace substring (oldValue) inside a string with new value.
Arguments	**String:** The string that you need to replace value, could be dynamic from controls. **oldValue:** The value that will be replaced by the new value, could be dynamic from controls. **newValue:** The value to replace the old value, could be dynamic from controls.
Argus Type	TextBox, Currency, TextArea, Rich Text Editor, Date, Time, DropDownList, Lookup, Label, HTMLText, Query Columns.
Return	String
Example	replace(TextBox1, 't', 'm')
Function Name	**split()**
Format	split(String, Separator, Limit)
Description	To split a string into an array of substrings.
Arguments	**String:** The string that you need to split it, could be dynamic from controls. **Separator:** Specifies the character to use for splitting the string, could be dynamic from controls. If omitted, the entire string will be returned (an array with only one item). **Limit (optional):** An integer that specifies the number of splits, it could be dynamic value from controls.

Argus Type	TextBox, Currency, TextArea, Rich Text Editor, Date, Time, DropDownList, Lookup, Label, HTMLText, Query Columns.
Return	Array
Example	split(TextBox1, ',', 2) (in case string was 'I,was', the first array will be "I" and the second one will be "was")
Function Name	**substring()**
Format	substring(String, Start, End)
Description	To extract the characters from a string, between two specified indices. It returns the substring of the first argument (String) starting at the position specified in the second argument (Start) and the length specified in the third argument (End).
Arguments	**String:** The string that you need to get its substring could be dynamic from controls values. **Start:** The position where to start the extraction. Must be Number. **End:** The position where to end the extraction. Must be Number.
Argus Type	TextBox, Currency, TextArea, Rich Text Editor, Date, Time, DropDownList, Lookup, Label, HTMLText, Query Columns.
Return	String
Example	substring(TextBox1, 2, 4) substring(TextBox1, 2, length(TextBox1))
Function Name	**toLower()**
Format	toLower(String)
Description	To convert the given string to lower case.
Arguments	**String:** The string that you need to convert, it could be dynamic from controls values.
Argus Type	TextBox, TextArea, Rich Text Editor, Date, Time, DropDownList, Lookup, Label, HTMLText, Query Columns.
Return	String
Example	toLower('SPARKnit')
Function Name	**toUpper()**
Format	toUpper(String)
Description	To convert the given string to upper case.
Arguments	**String:** The string that you need to convert, it could be dynamic from controls values.
Argus Type	TextBox, TextArea, Rich Text Editor, Date, Time, DropDownList, Lookup, Label, HTMLText, Query Columns.
Return	String
Example	toUpper('sparknit')
Function Name	**toUpperFirst()**
Format	toUpperFirst(String)
Description	To convert the given string to title case (first letter is capitalized).
Arguments	**String:** The string that you need to convert, it could be dynamic from controls values.
Argus Type	TextBox, TextArea, Rich Text Editor, Date, Time, DropDownList, Lookup, Label, HTMLText, Query Columns.
Return	String
Example	toUpperFirst('sparknit')
Function Name	**Trim()**
Format	trim(String)
Description	To remove any leading and trailing spaces from a string.
Arguments	**String:** The string that you need to trim, it could be dynamic from controls values.
Argus Type	TextBox, Currency, TextArea, Rich Text Editor, Date, Time, DropDownList, Lookup, Label, HTMLText, Query Columns.
Return	String
Example	trim('SPARKnit')
Function Name	**length()**

Format	length(String)
Description	To return the length of the given string.
Arguments	**String:** The string that you need to get its length, it can be dynamic value from controls.
Argus Type	TextBox, Currency, TextArea, Rich Text Editor, Date, Time, DropDownList, Lookup, Label, HTMLText, Query Columns.
Return	Number
Example	length('SPARKnit')

11.5 Users & Groups Functions

Function Name	**getUserEmail()**
Format	getUserEmail()
Description	To retrieve the current user email.
Arguments	None
Argus Type	None
Return	String
Example	getUserEmail()
Function Name	**getUserId()**
Format	getUserId()
Description	To retrieve the current user ID.
Arguments	None
Argus Type	None
Return	String
Example	getUserId()
Function Name	**getUserLogin()**
Format	getUserLogin()
Description	To retrieve the current user login.
Arguments	None
Argus Type	None
Return	String
Example	getUserLogin()
Function Name	**getUserName()**
Format	getUserName()
Description	To retrieve the current user name.
Arguments	None
Argus Type	None
Return	String
Example	getUserName()
Function Name	**getClaimLoginName()**
Format	getClaimLoginName()
Description	To retrieve the current user claim login.
Arguments	None
Argus Type	None
Return	String
Example	getClaimLoginName()
Function Name	**isGroupMemberById()**
Format	isGroupMemberById(groupId).
Description	Check if the current user is member in the given group Id.
Arguments	**groupId:** The group Id to check with if the current user is member in it.
Argus Type	None
Return	Boolean
Example	isGroupMemberById(1)
Function Name	**isGroupMemberByName()**
Format	isGroupMemberByName(groupName)

Description	Check if the current user is member in the given group name.
Arguments	**groupName:** The group name to check with if the current user is member in it.
Argus Type	None
Return	Boolean
Example	isGroupMemberByName('SPARK')
Function Name	**getCurrentUserProfileData()**
Format	getCurrentUserProfileData(nameOfProperties)
Description	Get user profile properties for the current user.
Arguments	An array containing the user profile properties which you want to get. If you leave it empty, it will get all properties for the current user.
Argus Type	None
Return	Array
Example	getCurrentUserProfileData (['AccountName', 'DisplayName', 'FirstName', 'CellPhone']);
Function Name	**getUserProfileData()**
Format	getUserProfileData(accountName, nameOfProperties)
Description	Get user profile properties for a specific user.
Arguments	**accountName:** The account name of a specific user. The accountName should be in the following format: 'Domain Name\Login name' if its static or it could be dynamic from form variable or form control. **nameOfProperties:** An array contains the user profile properties that want to get them. If leave it empty, you will get all the properties for the user.
Argus Type	None
Return	Array
Example	getUserProfileData('domain\Account', ['AccountName', 'DisplayName', 'FirstName', 'CellPhone']);

11.6 List Functions

Function Name	**updateListItems()**
Format	updateListItems(sitePath, listName, CAMLquery, fieldsToUpdate, newValues)
Description	To update items in a SharePoint list.
Arguments	**sitePath:** The list's site path. Insert single quotes to get the current site. **listName:** The name of the SharePoint list you need to update items. **CAMLquery:** The CAML Query. **fieldsToUpdate:** Array contains the internal names of current SharePoint list items columns you want to update them with new data. **newValues:** Array contains the new values that you want to replace old values with.
Argus Type	None
Return	Boolean
Example	updateListItems('/','Employee', "<Query><Where><Eq><FieldRef Name='ID'/><Value Type='Text'>10</Value></Eq></Where></Query>", ['Title', 'EmpName', 'EmpNo', 'JobTitle'],['Emp1', 'Jack', '1001', 'CEO'])
Function Name	**deleteListItems()**
Format	deleteListItems(sitePath, listName, CAMLquery)
Description	To delete items from a SharePoint list.

Arguments	**sitePath:** The list's site path. Insert single quotes to get the current site. **listName:** The name of the SharePoint list you need to delete items from. **CAMLquery:** The CAML Query.
Argus Type	Boolean
Return	None
Example	deleteListItems('http://server/','Employee', "<Query><Where><Eq><FieldRef Name='ID'/><Value Type='Text'>10</Value></Eq></Where></Query>")
Function Name	**newListItem()**
Format	newListItem(sitePath, listName, CAMLquery, fieldsToInsert, newValues, newSPFieldData, rootFolder)
Description	To add a new item into a SharePoint list.
Arguments	**sitePath:** The list's site path. Insert single quotes to get the current site. **listName:** The name of the SharePoint list you need to add the new item to. **fieldsToInsert:** Array contains the internal names of the current SharePoint list items columns you want to insert. **newValues:** Array contains the new values that you want to add to the list. In case the value is date or dateTime, you have to convert it into ISO format(). **newSPFieldData:** (Optional). Returns the newly created item SP column, such as the new item ID. If this parameter left empty the function will return Boolean value. **rootFolder:** [Optional] Specify the folder that the item related to. You can find the RootFolder from the URL. For example, suppose you have a folder named "IT" inside a list named "HR" which is located at the root site, then the rootFolder will be: "/Lists/HR/IT"
Argus Type	None
Return	Boolean
Example	newListItem('http://server/', 'Employee', ['Title', 'EmpName', 'EmpNo', 'JobTitle'], ['Emp2', 'Jack', '1002', 'CEO']) newListItem('http://server/', 'Employee', ['Title', 'EmpName', 'EmpNo', 'JobTitle'], ['Emp2', 'Jack', '1002', 'CEO'], '/Lists/HR/IT')
Function Name	**getListFieldData()**
Format	getListFieldData(sitePath, listName, FieldName, CAMLqueryString)
Description	To get values of a column from a list or library based on a specified CAMLQuery string
Arguments	**sitePath:** The site path. Set single quotes '' to get the current form's site path dynamically. **listName:** The name of the SharePoint list you need to get data from. **FieldName:** The internal name of the SharePoint list/library field you want to get its values. **CAMLqueryString:** CMAL Query string to be used to filter the list and get a specific list item's field value.
Argus Type	None
Return	Array
Example	getListFieldData('http://server/', 'SPARK List', 'Title', "<Query><Where><Eq><FieldRef Name='Title'/><Value Type='Text'>" + getValue(TextBox1)+ "</Value></Eq></Where></Query>");

See also

Reference

Other resources
Video: https://youtu.be/I7Z5XTExV7I. This video will show you how to add, update and delete items in other SharePoint lists

Function Name	**getListItemData()**
Format	getListItemData (sitePath, listName, CAMLquery, fieldToRetrevie, returnFlag)
Description	To retrieve values from a SharePoint list columns.
Arguments	**sitePath:** The site path. Set '' [two single quotes)] to get the current form site dynamically. **listName:** The name of the SharePoint list you need to get data from. **CAMLquery:** The CAML Query. **fieldToRetrevie:** Array contains the internal names of the current SharePoint list items fields along with their data. **returnFlag:** (optional) if you need to specify the returned value to be the ID or the Text or the combined value of a complex SharePoint column (such as lookup, Author, People Picker, External Data Picker …etc.). If you set this returnFlag to be blank/empty then the return value will be combined i.e. (ID#;Text) as stored in SharePoint. If the returnFlag is 'true' then it will return the ID value only, if returnFlag is 'false' then it will return the display text only.
Argus Type	None
Return	String

Example	var array= getListItemData('/', 'Services Profile', "<Query><Where><Eq><FieldRef Name='ID'/><Value Type='Text'>"+getValue(CertificateTypeLook)+ '</Value></Eq></Where></Query>', ['CertificateText', 'ApplicationFees', 'ServiceCode', 'ApplicationFeesCurrency'], false);
	
	CertificateTextVal=array[0];
	setValue(ApplicationFeesTxt, Math.round(array[1]));
	ServiceCode=array[2];
	setValue(ApplicationFeesCurrency, array[3]);

See also

Reference

Other resources
Video: https://youtu.be/-n6vUpbD4jA. This video will show you how to build and do integration between different lists within one form using getListItemData function.

Function Name	**getFieldsValues()**
Format	getFieldsValues(sitePath, listName, CAMLquery, fieldsToRetrieve, retrieveFormat)
Description	To retrieve data from a SharePoint list based on the retrieveFormat.
Arguments	**sitePath:** The site path. Set '' [two single quotes)] to get the current form site dynamically. **listName:** The name of the SharePoint list you need to get the data from. **CAMLquery:** The CAML Query. **fieldsToRetrieve:** Array contains the internal names of the current SharePoint list items fields along with their data. **RetrieveFormat:** Retrieve format takes one of the three options: onlyColumn, onlyRow, onlyField.
Argus Type	None

Return	Array
Example	getFieldsValues('http://server/', 'SPARK List', "<Query><Where><Eq><FieldRef Name='ID'/><Value Type = 'Text'>1</Value></Eq></Where></Query>", ['Title', 'EmpName'], 'onlyRow')

11.7 Miscellaneous Functions

Function Name	**getFormQueryString()**
Format	getFormQueryString(nameOfProperty)
Description	To retrieve the query string parameter value from the form URL.
Arguments	**nameOfProperty:** The required parameter name to get its value, this parameter is a part of the page URL, could be dynamic from controls.
Argus Type	TextBox, TextArea, Rich Text Editor, DropDownList, Lookup, Label, HTMLText
Return	String
Example	getFormQueryString('Title'), getFormQueryString(TextBox1)
Function Name	**addAttribute()**
Format	addAttribute(ControlId, attrName, Value)
Description	To add a new attribute inside the control.
Arguments	**ControlId:** The control id that you need to add the new attribute item inside. **attrName:** String contains the new attribute name you need to add, could be dynamic from controls. **Value:** String contains the new attribute value you need to add, could be dynamic from controls.
Argus Type	All
Return	None
Example	addAttribute(TextBox1, 'SPARKnit', 'SPARK')
Function Name	**modifyAttributeValue()**
Format	modifyAttributeValue(ControlId, attrName, Value)
Description	To modify attribute inside the control.
Arguments	**ControlId:** The control id that you need to modify the attribute item within. **attrName:** String contains the attribute name you need to modify, could be dynamic from controls. Value**:** String contains the new attribute value you need to add, could be dynamic from controls.
Argus Type	All
Return	None
Example	modifyAttributeValue(TextBox1, 'title', 'SPARK')
Function Name	**getAttributeValue()**
Format	getAttributeValue(ControlId, attrName)
Description	To retrieve attribute value inside the control.
Arguments	**ControlId:** The control id that you need to retrieve its attribute value. **attrName:** String contains the attribute name you need to retrieve its value, could be dynamic from controls.. **Value:** String contains the new attribute value you need to add, could be dynamic from controls.
Argus Type	All
Return	String
Example	getAttributeValue(TextBox1,'title')
Function Name	**clear()**
Format	clear(ControlId)
Description	To clear the given control value.
Arguments	**ControlId:** The control id that you want to clear it is value.

Argus Type	TextBox, Currency, TextArea, Rich Text Editor, Date, Time, DropDownList, Lookup, Label, Button, CheckBoxList, CheckBox, Image, HyperLink, PageViewer, External Data Dialog, People Picker, External Date Picker, Advanced Lookup, SQL Connector, Web Connector, XML Connector, HTMLText, Query Columns.
Return	None
Example	clear(TextBox1)
Function Name	**copyFromTo()**
Format	copyFromTo(ControlIdFrom, ControlIdTo)
Description	To copy a value from control to another control.
Arguments	**ControlIdFrom:** The control id that you want to copy value from. **ControlIdTo:** The control id that you want to copy value to.
Argus Type	TextBox, Currency, TextArea, Label (The Label can be used as ControlIdTo only)
Return	None
Example	copyFromTo(TextBox1, TextBox2)
Function Name	**getValue()**
Format	getValue(ControlId, LookupFlag)
Description	To get the given control value.
Arguments	**ControlId:** The control id that you want to return its value. **LookupFlag (Not Required):** Specify the returned Lookup value to be the ID or the Text. If you left this Flag empty or 'false' then the return value will be the ID of the lookup value. If the returnFlag is 'true' then it will only returns the text value,
Argus Type	TextBox, Currency, TextArea, Rich Text Editor, DropDownList, Label, Lookup, Date, Time, Button, Image, HyperLink, PageViewer, People Picker, Advanced Lookup, External Data Picker, External Data Dialog, SQL Connector, Web Connector, XML Connector, Query Columns.
Return	String
Example	getValue(TextBox1)
Function Name	**setFocus()**
Format	setFocus (ControlId)
Description	Sets the keyboard focus to the given control id.
Arguments	**ControlId:** The control id to set focus to.
Argus Type	TextBox, Currency, TextArea, RichBox, DateTime, Date, Time, DropDownList, Lookup, Button, CheckBoxList, CheckBox, HyperLink, PeoplePicker, ExternalPicker, ManagedMetadata
Return	None
Example	setFocus(TextBox1)
Function Name	**setValue()**
Format	setValue(ControlId, 'value')
Description	To set the control the given value.
Arguments	**ControlId:** The control id that you want to set the value to it. **Value:** The value that you want to set it in the control, could be dynamic from controls.
Argus Type	TextBox, TextArea, DropDownList, Label, Button, Lookup, Date, Time, Advanced Lookup, External Data Picker and External Data Dialog, Query Columns
Return	None
Example	setValue(TextBox1, 'SPARKnit') setValue(TextBox1, TextBox2)
Function Name	**show()**
Format	show(String)
Description	To show string or control value in an alert. The same as alert function, but you can specify a control to alert it is value.
Arguments	**String:** The string that you want to show it, could be dynamic from controls.

Argus Type	TextBox, Currency, TextArea, Rich Text Editor, Time, Date, DropDownList, Lookup, Label
Return	None
Example	show('SPARKnit') show(Textbox1)
Function Name	**getCurrentSitePath()**
Format	getCurrentSitePath()
Description	To retrieve the current site path name.
Arguments	None
Argus Type	None
Return	The current site pathname
Example	getCurrentSitePath()
Function Name	**getCurrentSiteUrl()**
Format	getCurrentSiteUrl()
Description	To retrieve the current site URL.
Arguments	None
Argus Type	None
Return	The current site URL
Example	getCurrentSiteUrl()
Function Name	**getWebServerRelativeURL()**
Format	getWebServerRelativeURL()
Description	Retrieve the current web server relative URL.
Arguments	None
Argus Type	None
Return	The current web server relative URL.
Example	alert(getWebServerRelativeURL()) //This will return "/spark" if the your form exists in "web server root/spark/…".

Function Name	**controlValueChanged()**
Format	controlValueChanged(ControlId)
Description	To check if a specific control has been triggered or its value has been changed, this function is mainly used in the condition part of the rule in order to execute the action or apply the format depending on triggering that specific control. This function is used in general to prevent changing on a calculated or auto-generated value unless the change or the trigger occurs on the specific control.
Arguments	**ControlId:** The control id that you want to check if it has been triggered or changed. If you leave it empty, then the function will check for the control that you are creating the rule on.
Argus Type	TextBox, Currency, TextArea, Rich Text Editor, DropDownList, Lookup, Date, Time, DateTime, People Picker, Advanced Lookup, External Data Picker, External Data Dialog, SQL Connector, Web Connector, XML Connector, Checkbox, CheckBoxList,RadioButton
Return	True if the given control or current control has been triggered or changed, otherwise it returns false
Example	controlValueChanged(), controlValueChanged(TextBox1)
Function Name	**controlValueOnLoad()**
Format	controlValueOnLoad(ControlId)
Description	To check if a specific control has been triggered or its value has been changed when loading the form. This function is mainly used in the condition part of the rule in order to execute the action or apply the format depending on triggering the specific control when loading the form.
Arguments	**ControlId:** The control id that you want to check if it is value has been triggered or changed when loading the form, if you leave it empty, then the function will check for the control you are creating

	the rule on.
Argus Type	TextBox, Currency, TextArea, Rich Text Box, DropDownList, Lookup, Date, Time, DateTime, People Picker,Advanced Lookup, External Data Picker, External Data Dialog, SQLConnector,WebConnector,XMLConnector, CheckBox, CheckBoxList, RadioButton.
Return	Boolean. True if the given control or current control has been loaded, otherwise it returns false.
Example	controlValueOnLoad() controlValueOnLoad(TextBox1)
Function Name	**closeDialog()**
Format	closeDialog(controlID)
Description	To close any open dialog.
Arguments	**controlID:** The id of the control or view that want to close the related dialog.
Argus Type	None
Return	None
Example	closeDialog(Panel9) closeDialog('Panel9') closeDialog('viewName')
Function Name	**openDialog()**
Format	openDialog(controlID, title, width, height)
Description	Open an embedded popup dialog from within the form to display specific panel or controls set in that dialog such as list or any informative data.
Arguments	**controlID:** The Control ID which you want to be displayed in the dialog, it can be a grouping control such as panel, Tab or any other single control ID. **Title** (optional)**:** The title of dialog. **Width** (optional)**:** The dialog width, the default value is 400. **Height** (optional)**:** The dialog height, the default value is 400.
Argus Type	None
Return	None
Example	openDialog(Panel9, 'Test Dialog', 427, 492)

See also

Reference

Other resources

Video: https://youtu.be/pcgxZxo7ex4. This video will show you how easily to create or deal with popup dialog in your forms dynamically.

11.8 Calculated Columns Functions

Function Name	**abs()**
Format	abs(Number)
Description	Returns the absolute value of a given number.
Arguments	**Number:** The number whose absolute value is to be.
Argus Type	None
Return	Number
Example	abs(-5)
Function Name	**addYears()**
Format	addYears(DateTime, Number)
Description	To add or subtract number of years to date or datetime.
Arguments	**DateTime:** The value of date, datetime. It could be dynamic from control. **Number:** The number of years that want to add or subtract from the given DateTime. Negative or positive integer value.

Argus Type	None
Return	New date
Example	addYears(\"10/20/2018\",1) , addYears(Date1,2)
Function Name	**addMonths()**
Format	addMonths(DateTime, Number)
Description	To add or subtract number of months to date or datetime.
Arguments	**DateTime:** The value of date, datetime. It could be dynamic from control. **Number:** The number of months that want to add or subtract from the given DateTime. Negative or positive integer value.
Argus Type	None
Return	New date
Example	addMonths(\"10/20/2018\",1) , addMonths(Date1,2)
Function Name	**addDays()**
Format	addDays(DateTime, Number)
Description	To add or subtract number of days to date or datetime.
Arguments	**DateTime:** The value of date, datetime. It could be dynamic from control. **Number:** The number of days that want to add or subtract from the given DateTime. Negative or positive integer value.
Argus Type	None
Return	New date
Example	addDays(\"10/20/2018\",1) , addDays(Date1,2)
Function Name	**addHours()**
Format	addHours(DateTime, Number)
Description	To add or subtract number of Hours to datetime.
Arguments	**DateTime:** The value of datetime. It could be dynamic from control. **Number:** The number of Hours that want to add or subtract from the given DateTime. Negative or positive integer value.
Argus Type	None
Return	New date
Example	addHours(Date1,2)
Function Name	**addMinutes()**
Format	addMinutes(DateTime, Number)
Description	To add or subtract number of minutes to datetime.
Arguments	**DateTime:** The value of datetime. It could be dynamic from control. **Number:** The number of minutes that want to add or subtract from the given DateTime. Negative or positive integer value.
Argus Type	None
Return	String
Example	addMinutes(Date1,2)
Function Name	**dateDiffYears()**
Format	dateDiffYears(StartDateTime, EndDateTime)
Description	Determines the number of years between two dates
Arguments	**StartDateTime:** The value of start date. **EndDateTime:** The value of end date.
Argus Type	None
Return	Number
Example	dateDiffYears('10/10/2018','10/10/2016'); will return 2.
Function Name	**dateDiffMonths ()**
Format	dateDiffMonths (StartDateTime, EndDateTime)
Description	Determines the number of months between two dates
Arguments	**StartDateTime:** The value of start date. **EndDateTime:** The value of end date.
Argus Type	None
Return	Number
Example	dateDiffMonths('08/10/2018','10/10/2018'); will return 2.

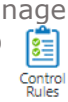
Function Name	**dateDiffDays()**
Format	dateDiffDays(StartDateTime, EndDateTime)
Description	Determines the number of days between two dates
Arguments	**StartDateTime:** The value of start date. **EndDateTime:** The value of end date.
Argus Type	None
Return	Number
Example	dateDiffDays('10/20/2018','10/21/2018') ; will return 1.
Function Name	**dateDiffHours()**
Format	dateDiffHours(StartDateTime, EndDateTime).
Description	Determines the number of hours between two dates
Arguments	**StartDateTime:** The value of start date. **EndDateTime:** The value of end date.
Argus Type	None
Return	Number
Example	dateDiffHours('10/20/2018 10:40:20 pm','10/20/2018 11:40:20 pm'); will return 1
Function Name	**dateDiffMinutes()**
Format	dateDiffMinutes(StartDateTime, EndDateTime)
Description	Determines the number of minutes between two dates
Arguments	**StartDateTime:** The value of start date. **EndDateTime:** The value of end date.
Argus Type	None
Return	Number
Example	dateDiffMinutes('10/20/2018 11:30:20 pm','10/20/2018 11:40:20 pm'); will return 10.
Function Name	**dateDiffSeconds()**
Format	dateDiffSeconds(StartDateTime, EndDateTime)
Description	Determines the number of seconds between two dates
Arguments	**StartDateTime:** The value of start date. **EndDateTime:** The value of end date.
Argus Type	None
Return	Number
Example	dateDiffSeconds('10/20/2018 11:40:20 pm','10/20/2018 11:40:30 pm'); will return 10.
Function Name	**year()**
Format	year(DateValue)
Description	To extract year from a specific Date or DateTime.
Arguments	**DateValue:** Specify the date value to extract the year from it. It could be dynamic from control.
Argus Type	None
Return	Number
Example	year('11/25/2018')
Function Name	**month()**
Format	month(DateValue)
Description	To extract month from a specific Date or DateTime.
Arguments	**DateValue:** Specify the date value to extract the month from it. It could be dynamic from control.
Argus Type	None
Return	Number
Example	month('11/25/2018')
Function Name	**dayOfWeek()**
Format	dayOfWeek(DateValue)
Description	To extract day of the week from a specific Date or DateTime.
Arguments	**DateValue:** Specify the date value to extract the day of the week from it. It could be dynamic from control.
Argus Type	None

Return	Number
Example	dayOfWeek('11/25/2018')
Function Name	**toUniversalTime()**
Format	toUniversalTime(DateValue)
Description	To extract universal datetime value from a specific Date or DateTime.
Arguments	**DateValue:** Specify the date value to extract the universal datetime value from it. It could be dynamic from control.
Argus Type	None
Return	ISO Date
Example	ToUniversalTime('11/25/2018')
Function Name	**time()**
Format	time(DateValue)
Description	To extract time from a specific DateTime.
Arguments	**DateValue:** Specify the date value to extract the time from it. It could be dynamic from control.
Argus Type	None
Return	Time
Example	time('11/25/2018 11:30:20 pm')
Function Name	**power()**
Format	power(Number, powerOfNumber)
Description	Raises a number to the specified power.
Arguments	**Number:** The number that you want to power. It could be dynamic from control. **powerOfNumber:** Specify the power value. It could be dynamic from control.
Argus Type	None
Return	Number
Example	power(4,3); will return 64
Function Name	**round()**
Format	round(Number)
Description	Rounds a decimal value to the nearest integer.
Arguments	**Number:** Decimal value that you want to round. It could be dynamic from control.
Argus Type	None
Return	Number
Example	round(4.7); will return 5
Function Name	**formatDate()**
Format	formatDate(DateValue,Format)
Description	Represents a date time value in text of a specific format.
Arguments	**DateValue:** Specify the date value to format. It could be dynamic from control. **Format**: Date Format that want to apply on the DateValue.
Argus Type	None
Return	Date
Example	formatDate(\"10/20/2018\",\"MM-dd-yyyy\"); will return 10-20-2018
Example	getCurrentSitePath()

12 Rules

A rule can be used to add dynamic formatting, action, formula or validation commands/scripts that affect controls within the report and Ad-Hoc query form at runtime. In addition, rules can be applied on the report or Ad-Hoc query form directly, same as with controls.

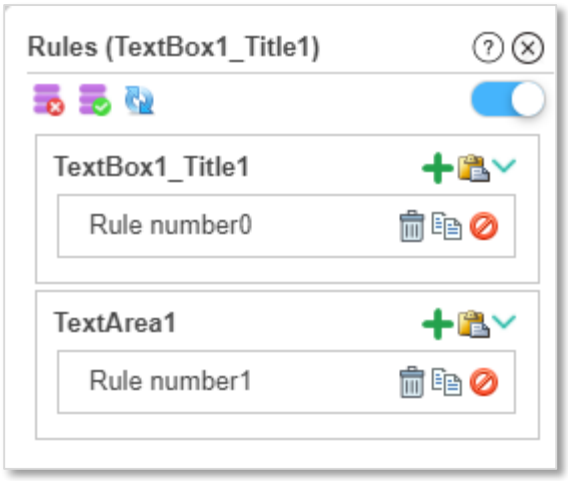
12.1 Opening the Rules Panel

To open the a control or the Reports rules panel, select the control you need to create or manage its rules, you will see that the Control Rules button appear in the control ribbon tab 

click on it to create/manage control-based Rules. The Rules manage panel will appears at the left side of the report design workspace. You can click on **Report Rules** button 

to create/manage report-based rules as well.

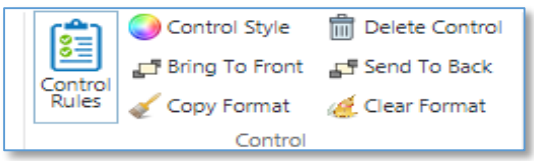
By default, the rules panel will show only those rules that are associated with every control or report currently selected. The rules panel has the following buttons in order to work with the rules associated with the selected control, these buttons are:



- **Show all:** By checking this box, you can view all related rules of the report and its controls. Click on the rule will open the Rule Manager.
- **Create:** Create a new rule on the selected control.
- **Disable/Enable:** Disable or enable the selected rule on the selected control. If the selected rule is disabled, the enable icon will be shown, If the selected rule is enabled, the disable icon will be shown. Note that disabled rules will not be executed at runtime.
- **Enable All:** Enable all rules on the selected control.
- **Disable All:** Disabled all rules on the selected control.
- **Copy Rule:** Copy the selected rule for the control (you need to select a rule in order to enable this button).
- **Paste Rule:** Paste the copied rule from a control to the selected control.
- **Refresh:** Refresh rules list in the selected control.
- **Delete Rule:** Delete the selected rule (you need to select a rule in order to enable this button).
- **Move Rules up and down (Priority):** You can move the rules for the selected control up and down by clicking on the rule and moving the mouse cursor up and down while pressing the mouse right key down. Uppers rules have higher priority than the down ones (you need to select a rule in order to enable this button).

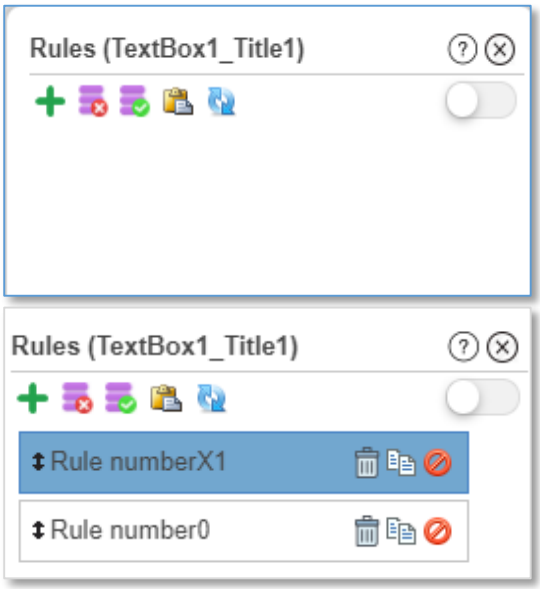
12.2 Adding Rules

1. Select the control or the report you want to create/manage its rules.
2. Click the **Control Rules** button in the **Control Group**, or the **Report Rules** button in the under the **Settings tab** ribbon.



3. The Rules pane will appear at the left side of the report designer.
4. Press the Create a rule button ✚.
5. The newly created rule will be assigned to the selected control or report and will appear in the rules pane list.

Note: Once you created and save a rule, the rule will be automatically active (enabled). To delete a rule, click on the rule and click on the **Delete** icon. To disable a rule, click on the rule then the **Disabled** icon.

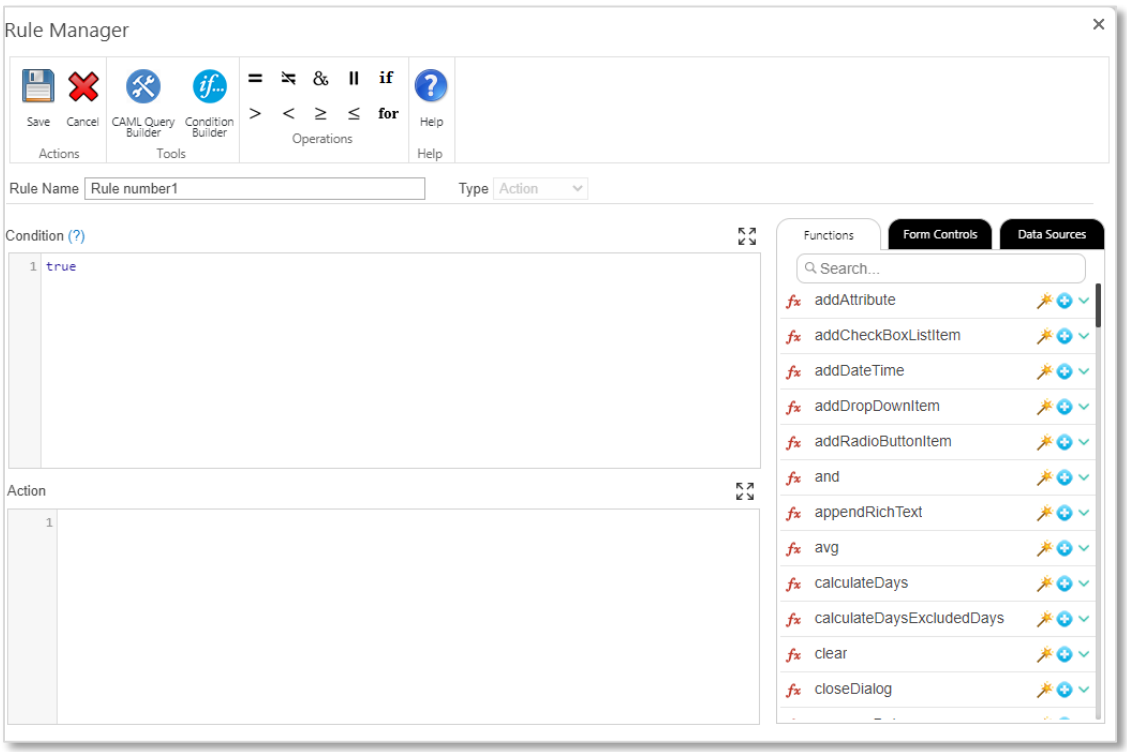


12.3 Editing Rules

To edit a rule, click on it in the rules pane list, edit the rule as desired and click save.

12.4 Rule Manager

The **Rule Manager** pane appears when you edit or create a rule, to add a rule, you can click on the ✚ button and to edit a rule, you have to double click on the desired rule in the rules list, edit the rule as desired and click save button.



12.4.1 Rule Manager Overview

The Rule Manager Ribbon

- **Save:** Save or updates the rule settings.
- **Cancel:** Clicking **Cancel** will close the **Rules Manager** and return to the original location.

- **CAML Query Builder:** For more details, click on CAML Query Builder.
- **Condition Builder:** For more details, click on Condition Builder.
- **Operations:** This group has the standard logic operations symbols that can be used to evaluate, validate or compare values in order to create conditions and/or scripts.
  String literals must be contained in double quotes (e.g. TextBox1=="High").

Icon	Symbol	Meaning
=	==	equal to
≠	!=	not equal to
>	>	greater than
<	<	less than
≥	>=	greater than or equal to
≤	<=	less than or equal to
&	&&	And
||	||	Or

if	If statement. The syntax is:
	if(/*Condition*/){
	//Code
	}else{
	//Else Code }	

for	JS For statement…	

Rule Name:
The rule name is the identity of the rule, which represents the rule function in the form, rule name must be unique per control and the "Rule Manager" will not allow you to duplicate the rules names.

Rule Type:
The rule types categorized into four main categories for Ad-Hoc query form's controls and canvas: **Validation**, **Formatting**, **Action**, **Ready Rule** the default is the validation type. For Report's canvas and controls, there are two main rules types: Action and Formatting. For Report Footer Cells, there are three main rules types: **Formula**, **Action** and **Formatting**:

Note: Formatting and Ready rules are not applicable on report and AD-Hoc query form's canvas.

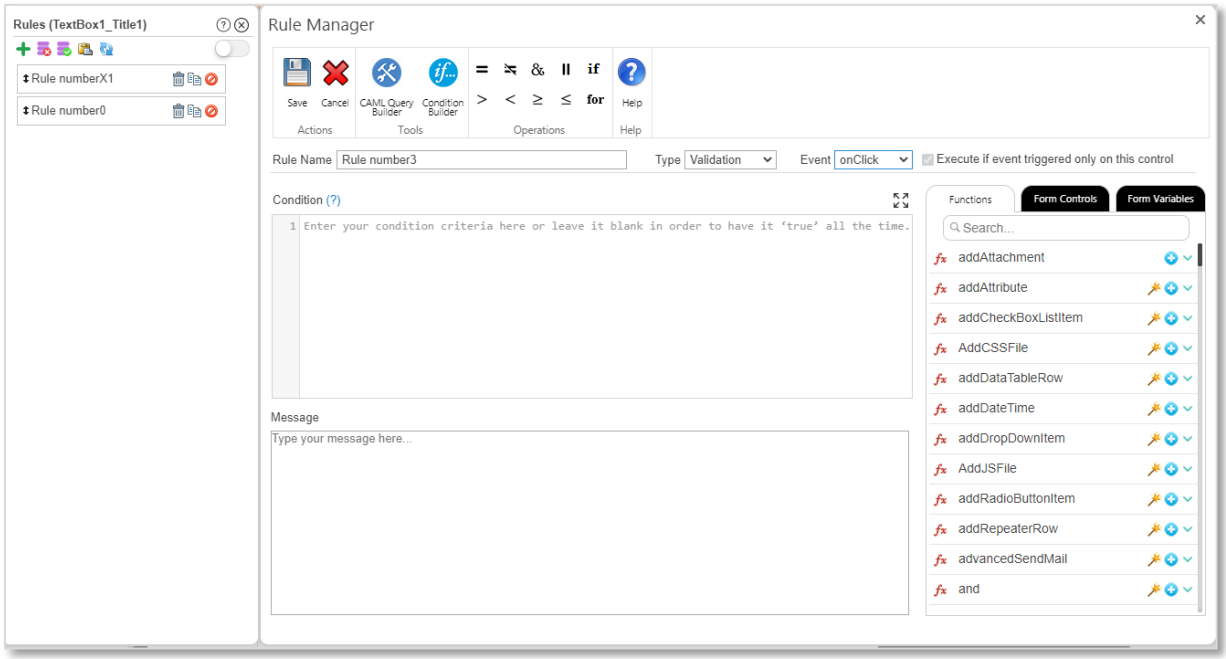
Rule Events:
Applicable only on Ad-Hoc Query Form's controls.
Contains events types list, which you can select from, to trigger the rule based on the selected type. The following describes in details the events types:

- **onLoad:** The onLoad event executes the rule only one time when the form loads.

- **onChange:** The onChange event executes the rule every time a value of the control changes.
- **onKeyup:** The onKeyup event executes the rule every time a keyboard key pressed and released on a control.
- **onClick:** The onClick event executes the rule every time a user clicks on the control.

12.4.2 Validation Rule

Validation type is usually selected when you need to add a certain validation rule to a control or a form. The validation consists of two main inputs (condition and message); you can leave the condition part empty in case you want this validation to run when the form loads.



Condition: This input is to mark the control as invalid and to prevent executing the Ad-Hoc form (When clicking the Run Report Button). If the condition expression evaluates to **true**. If the expression evaluates to **false**, the rule will not be triggered. The Assistance Panel (to the right of the Rule Manager) can be used to assist you constructing the rule's condition formulas, operations, actions and validations for the controls and the form as well. For more details, click on Assistance Panel.

The Ad-Hoc form will not execute if the validation rule evaluates to "**true**".

The rule should be applied to the control for which you want to highlight the issue. If the rule does not evaluate, a red box will appear around the relevant control

Note: You can write your condition in the condition text box. In order to write a correct condition syntax do not include (if statement) in the syntax, have a look at the following examples:

Example 1:TextBox1 == 0
Example 2:Checkbox1 == true
Example 3:Checkbox1 == true && TextBox1 == 0

You can include any JS or JQ syntax in the condition area, for example:

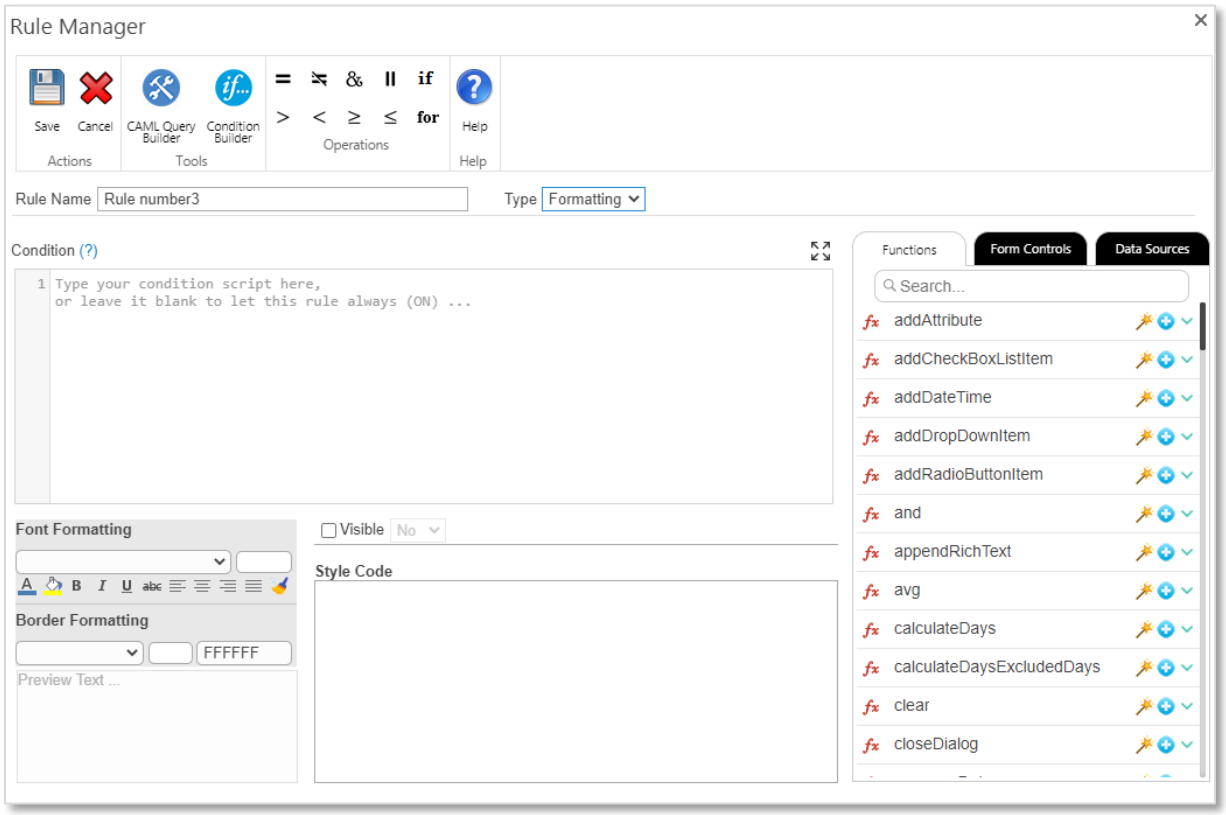
if (Checkbox1 == true);

Message: The message to be displayed when the validation rule is triggered.

Conditional Validation Rule Example:
A form has two input controls, a textbox, which is called "Amount" and a multi-line text control called "Clarification". A validation rule is added to the "Clarification" control with the condition "Amount>1000&&isEmpty(Clarification)". If a user enters an amount greater than 1000 in the form, he/she cannot submit the form unless he justifies the expenses in the "Clarification" control.

12.4.3 Formatting Rules

Formatting type is usually selected when you need to change the style formatting of the selected control, the formatting is usually used to affect the control style, visibility, make it read-only or/and disable it.



Condition: This input is set the condition script in order to change the selected control style format, visibility or behavior when this expression evaluates to **True**. If the expression evaluates to **False**, the rule will not be triggered and the control format will not change. The Assistance Panel (to the right of the Rule Manager) can be used to assist you constructing the rule's condition formulas, operations, actions and validations for the controls and the form as well. For more details, click on Assistance Panel.

Note: You can write your condition in the condition text box. In order to write a correct condition syntax do not include (if statement) in the syntax, have a look at the following examples:

Example1: TextBox1 == 0
Example2: Checkbox1 == true
Example3: Checkbox1 == true && TextBox1 == 0

You can include any JS or JQ syntax in the condition area, for example:

if (Checkbox1 == true);

The following symbols can be used as comparison operators. String literals must be contained in double quotes (e.g. Option=="High").

Symbol	Meaning
==	equal to
!=	not equal to

Symbol	Meaning
>	greater than
<	less than
>=	greater than or equal to
<=	less than or equal to
&&	and
||	Or

Formatting: You can use the Style Manager inside the formatting rule to change the selected control style; the "**Style Manager**" consists of the following buttons:

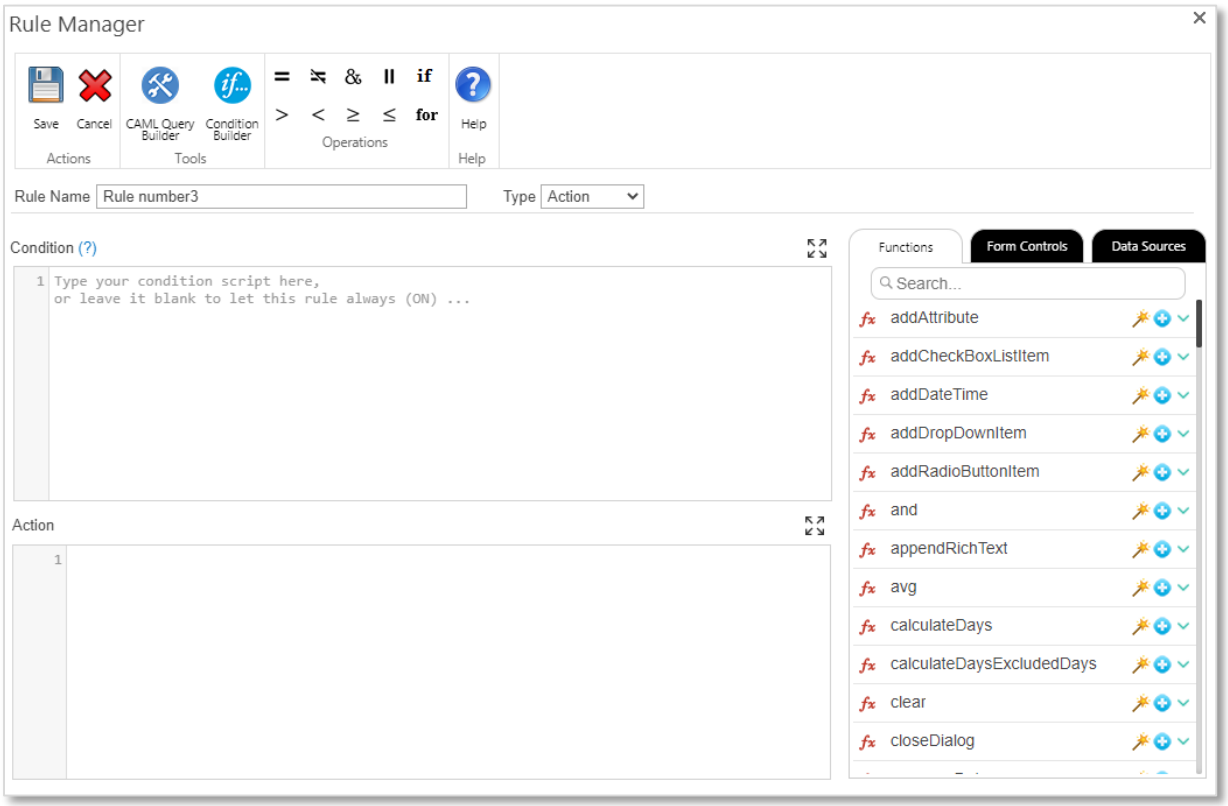
- **Font Family:** Change the font family of the control.
- **Font Size:** Change the font size of the control.
- **Fore color:** Change the font color of the control.
- **Back color:** Set the background Color of the text.
- **Emphasis (B bold, U underline, I italics and abc strikethrough):** Set the text to bold, underline, italics and strikethrough.
- **Align text:** your can align text to left, center right or justified text.
- **Clear formatting:** Remove the formatting added of the control
- **Border Style:** Apply border style of the selected control as solid/dashed/..etc.
- **Border Width:** Set the border width of the control (Numeric value).
- **Border Color:** Set the border color of the control
- **Preview:** This area will preview how the control style would appear if the rule applies on the control.
- **Style Code:** A free code input, enabling the user to write his own style script for the selected control.

Control Behavior Pane:

- **Visible:** Select Yes or No values in order to show or hide the selected control based on the condition expression if evaluates to true.
- **Disabled:** Select Yes or No values in order to be used to disable or enable the control based on the condition expression if evaluates to true.
- **ReadOnly:** Select Yes or No values in order to be used to change the selected control behavior as a read-only input or not based on the condition expression if evaluates to true, this option is usually used for input controls.

12.4.4 Action Rules

The Action type is usually selected when you need to add an action rule to a control or a form. The action rule consists of two main inputs (Condition and Action); you can leave the condition part empty in case you want this action to run when the form loads.



Condition: Where the designer can write his condition script in order to execute the action when the condition is met (**true)**. If the condition is **false** then the rule will not be executed "triggered" and no action will be occurred. The Assistance Panel (to the right of the Rule Manager) can be used to assist you constructing the rule's condition formulas, operations, actions and validations for the controls and the form as well. For more details, click on Assistance Panel.

The rule should be applied to the control and will be triggered on the (onChange, onKeyup) depending on the control type.

Note: You can write your condition in the condition text box. In order to write correct condition syntax do not include (if statement) in the syntax, have a look at the following examples:

Example 1:TextBox1 == 0
Example 2:Checkbox1 == true
Example 3:Checkbox1 == true && TextBox1 == 0

You can include any JS or JQ syntax in the condition area, for example:

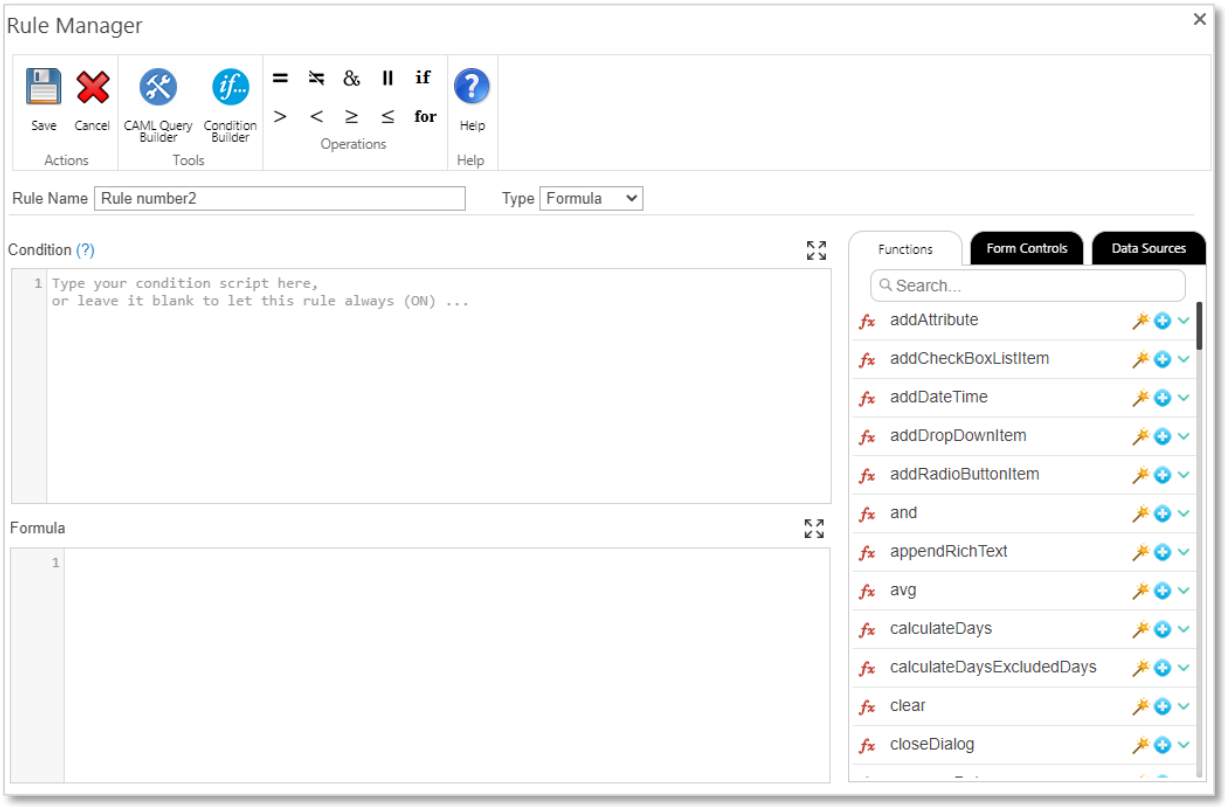
if (Checkbox1 == true);

Action: This input is to have the action script inside, the script could be ready built-in functions (refer to the Assistance Panel for more details), JQ or JS code, and will be executed on the (onChange, onKeyup, onLoad, onSave) events depending on the selected control type and only when the condition expression is **true**.

Example: setValue(TextBox1,"Hello World")

12.4.5 Formula Rules

The Formula type is only available on **Formula Control, Report Table Columns and Report Table Footer Cells**, which is usually selected when you need to generate/execute a formula script on these types of controls. The Formula rule consists of two main inputs (Condition and Formula); you can leave the condition part empty in case you want this formula to run when the report loads.



Condition: Where the designer can write his condition script in order to execute the formula when the condition is met (**true)**. If the condition is **false** then the rule will not be executed "triggered" and no action will be occurred. The Assistance Panel (to the right of the Rule Manager) can be used to assist you constructing the rule's condition formulas, operations, actions and validations for the controls and the form as well. For more details, click on **Assistance Panel**.

Note: You can write your condition in the condition text box. In order to write correct condition syntax do not include (if statement) in the syntax, have a look at the following examples:

Example 1:TextBox1 == 0
Example 2:Checkbox1 == true
Example 3:Checkbox1 == true && TextBox1 == 0

You can include any JS or JQ syntax in the condition area, for example:

if (Checkbox1 == true);

Formula: This input is to have the formula script inside, the script could be ready built-in functions (refer to the **Assistance Panel** for more details), JQ or JS code.

Example:(Sales1000__SPR__0_Sale_x0020_Value - Sales1000__SPR__0_Discount) * 8

Notes:

- We recommend using the **Assistance Panel** to get the Data Source columns names in the formula section, as their internal names are complex.
- The **Formula Rule** must return a value to be set in the control; otherwise, it will be considered as an invalid rule.

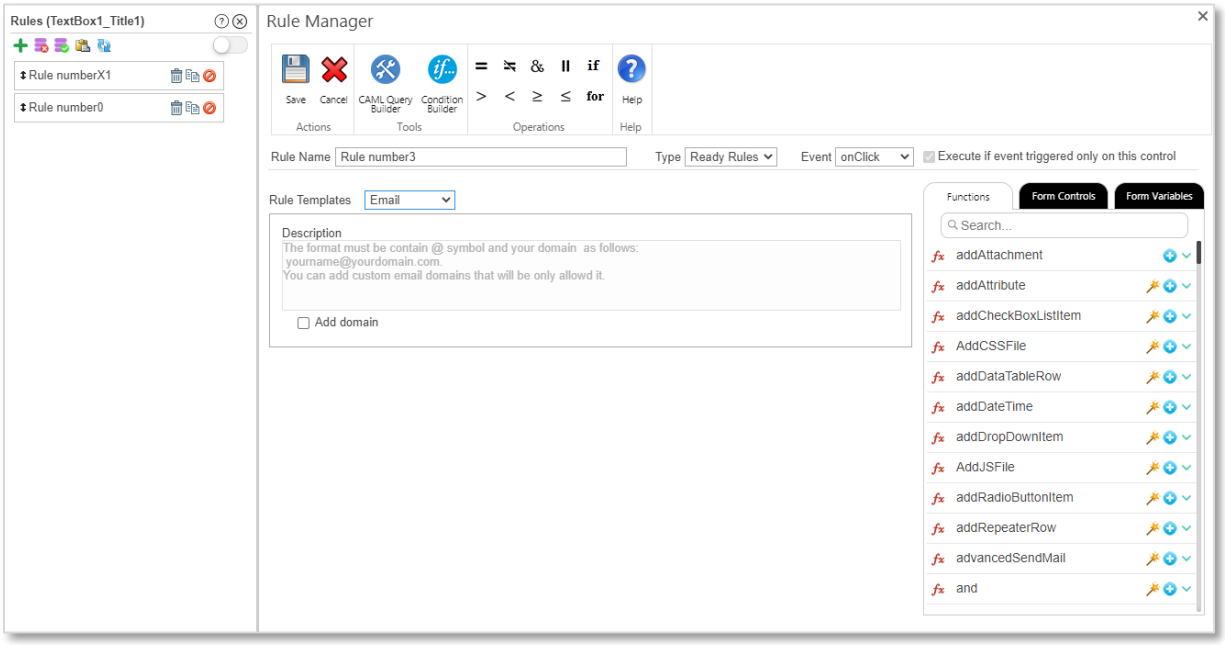
12.4.6 Ready Rules

The ready rules are ready rule's functions, which can be applied on a control without the need to do any coding from the designer side.

Rule Templates

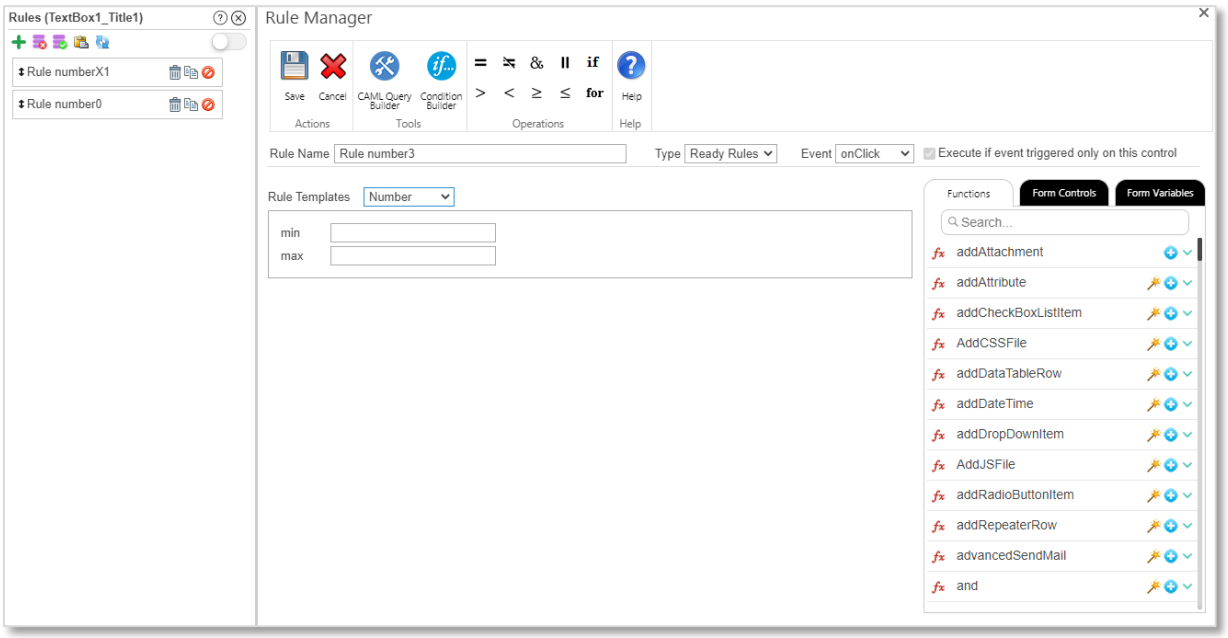
There are four ready rules that you can apply them (Email, Number, Input Length, and Pattern), you can select anyone of them from the dropdown list. See below the description for every one of them:

- **Email:**



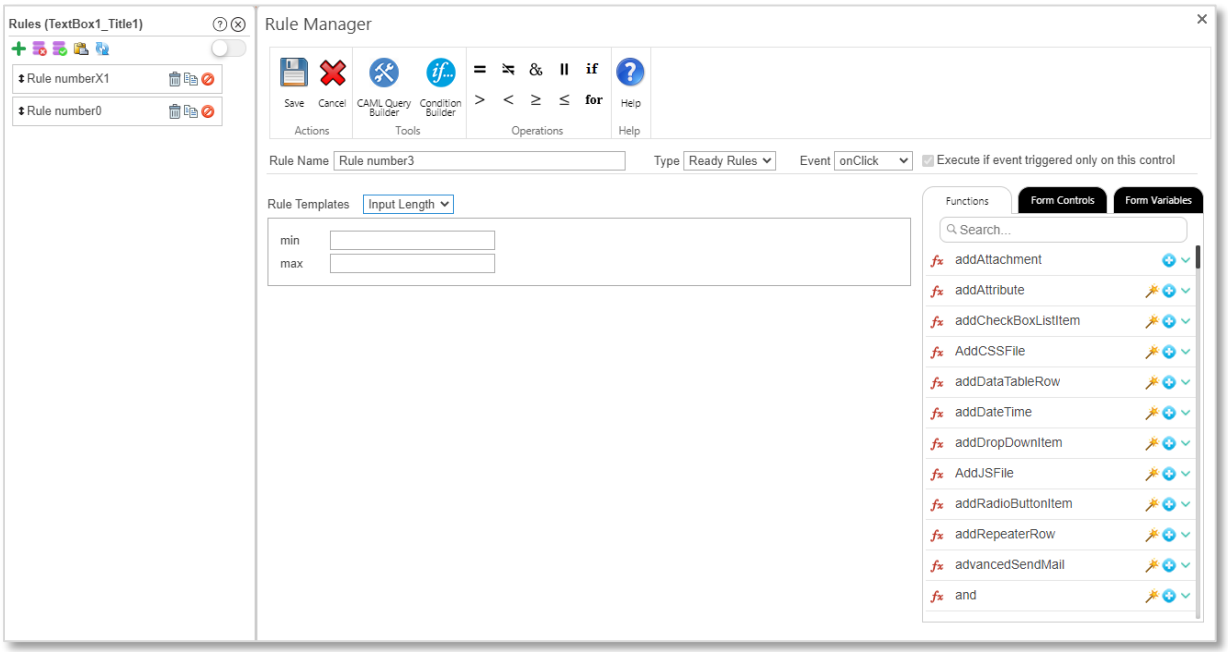
This rule template is for applying special validation rule for the input control, to validate if the type email is valid or not, also you could add a set of domains which the email must be belonging to in order to the condition to be met.

- **Number:**



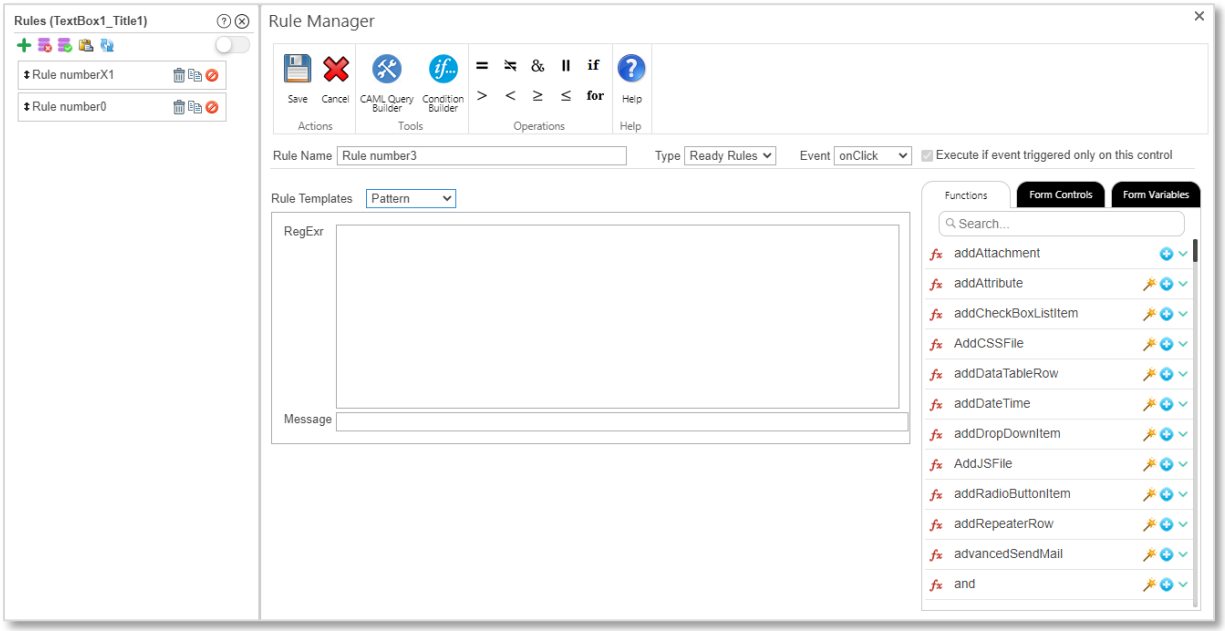
This rule template is for applying a special validation rule for the input control, to validate if the type of data inserted in an input control is numbers and you could specify the minimum and maximum range too.

- **Input Length:**



This rule template is for applying a special validation rule for the input control, to validate if the number of characters of the data inserted in an input control will not exceed the range.

- **Pattern:**



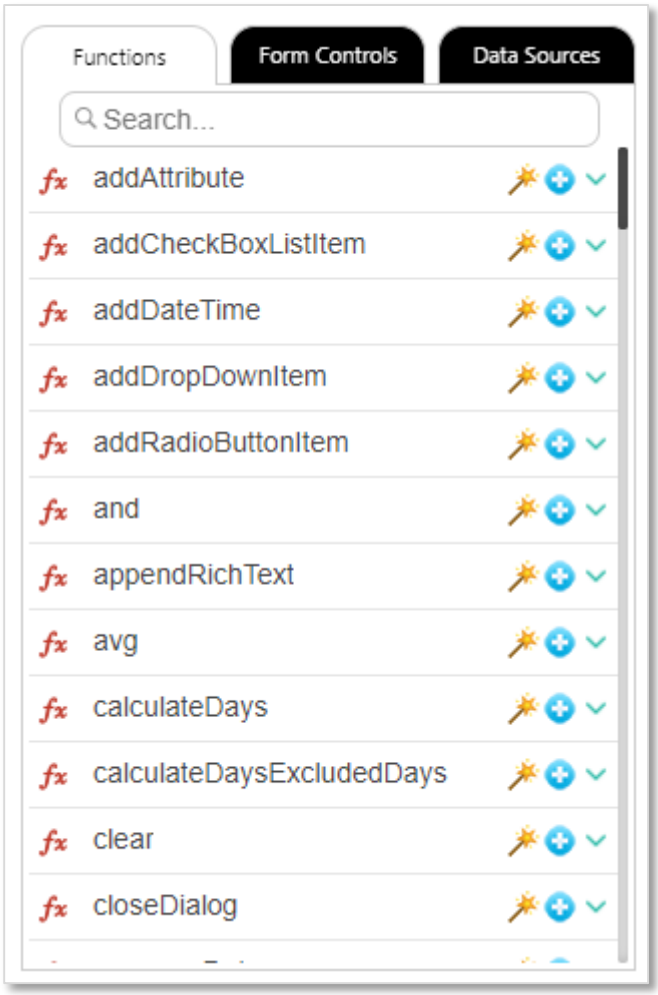
This rule template is for applying a special validation rule for the input control, to validate if the data inserted in an input control against the regular expression formula (Pattern) specified in the (RegExr) input is true. If the data inserted in the input control against the regular expression is false, the specified message will be shown when the mouse cursor hovers over the control.

The below table provides examples of a popular regular expressions. For more details about regular expressions, please refer to https://regexr.com/ and https://www.rexegg.com/.

Usage Example	Regex Pattern	String That Match	String That Doesn't Match
Match a Username	/^[a-z0-9_-]{3,16}$/	my-us3r_n4m3	th1s1s-wayt00_l0ngt0beausername (its too long)
Match a Password	/^[a-z0-9_-]{6,18}$/	myp4ssw0rd	mypa$$w0rd (it contains a dollar sign)
Match an Email	/^([a-z0-9_\.-]+)@([\da-z\.-]+)\.([a-z\.]{2,6})$/	john@doe.com	john@doe.something (the TLD is too long)
Match Any Email Address from a Specific Domain	^[a-zA-Z0-9_.+-]+@(?:(?:[a-zA-Z0-9-]+\.)?[a-zA-Z]+\.)?(domain1|domain2)\.com$	john@domain1.com	john@domain3.com
Math any IP address	^(([0-9]|[1-9][0-9]|1[0-9]{2}|2[0-4][0-9]|25[0-5])\.){3}([0-9]|[1-9][0-9]|1[0-9]{2}|2[0-4][0-9]|25[0-5])$	192.168.1.1	192.168.1

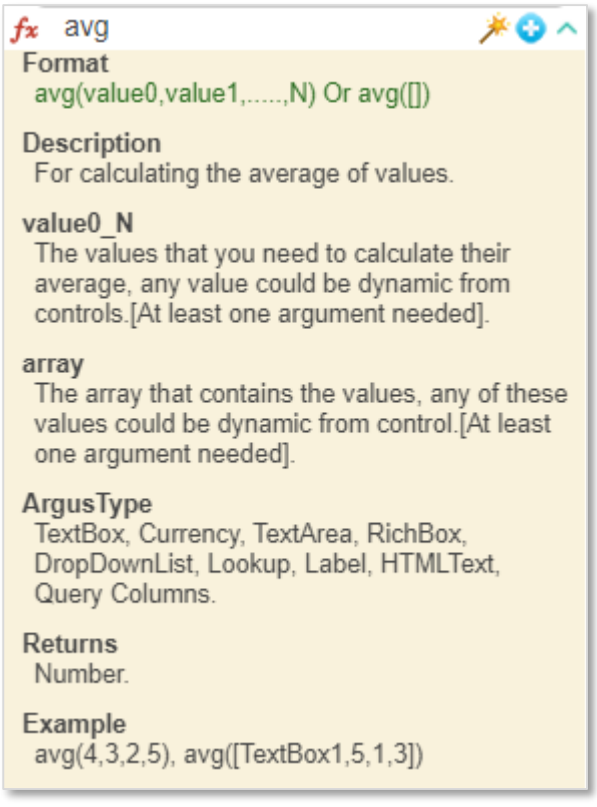
12.4.7 Assistance Panel

Assistance Panel is usually used to assist you constructing the rule's condition formulas, operations, actions and validations for the controls and the report's as well; it provides built in functions online help including descriptions and examples for each function when clicking on the scroll down information icon .
The Assistant Panel consists of three main tabs (Functions, Controls, and Data Sources).



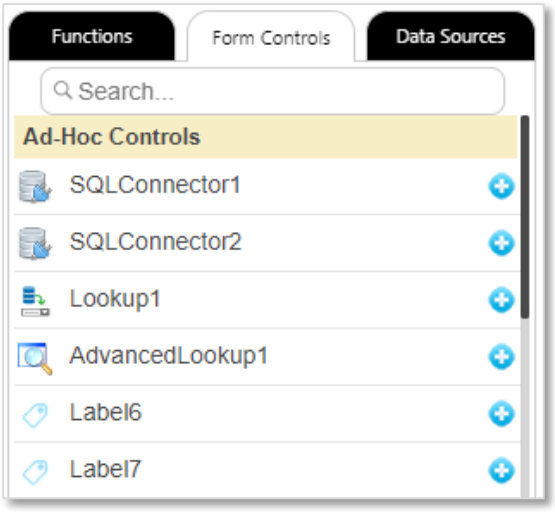
Functions:
This tab has a wide collection of ready-made functions created by SPARK Report's engineers, to make it easy for users to implement the needed functionalities in the report and to reduce the time needed to implement them from scratch. New functions will be added in every release of SPARK Reports Builder. You can add a function to the condition or action parts of the rule by a double click on the function name or click on the insert function icon .
To show the description of the function, click on the icon next the name of the function to scroll down the description information and details. **Note: Functions**

Description shows the description of each function selected in the functions list, which includes the following information: (Format, Description, Function Variables, Arguments Type, Returns and Examples).

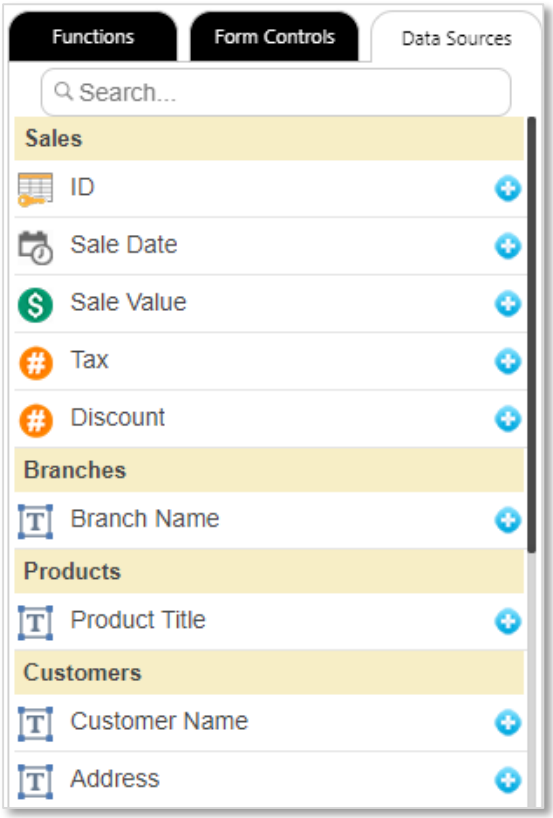


In addition, there is a **Function Wizard (Case Manager)** icon to help you easily build a script for some complex functions. When clicking on the Function Wizard icon, the Case Manager Wizard will open to assist you in building the script by displaying several wizard's dialogs; these dialogs will guide you in a systematic method to create your scripts without the need to do it manually.

Form Controls:
This tab has all controls IDs available in the designed form. You include any control in your scripts by simply clicking on it in this tab or by clicking on the insert control icon . For instance, you can include these controls in (conditions, actions and validations) by double clicking the control ID in the tab, this will reference the value of the control associated to a rule within a condition or in the Action pane. For instance, set a rule's condition to "Checkbox1 == true", set an action rule "getValue(TextArea1) ".



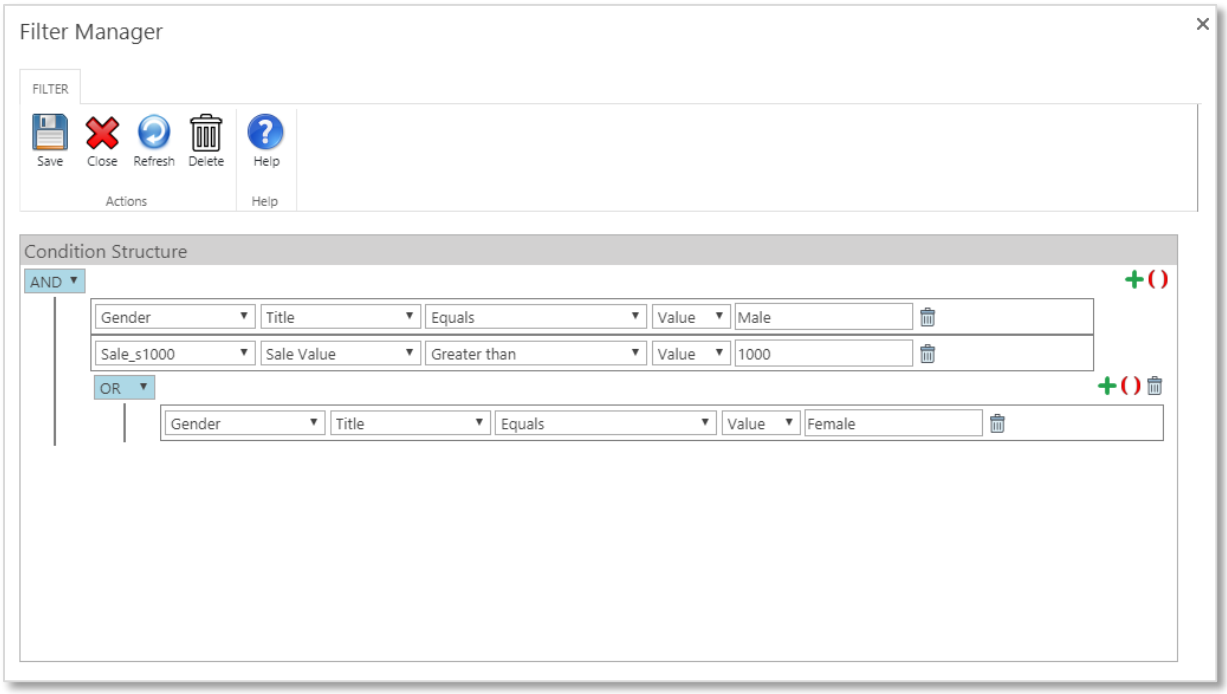
Data Sources:
This tab has all report's query data sources selected and calculated columns. Through this tab you can include any data source's column in your scripts (conditions, actions, formulas and validations) by double clicking the column name in the list or by clicking on the insert control icon , to get its value in the rule's script/condition at runtime.



13 Filters Manager

The **Filters Manager** is a user interface that helps query's designers building simple/ complex and static/dynamic filters for the query.

13.1 Filters Manager Ribbon



- **Save:** Save the **Filters Manager** designed criteria. Note that the filter criteria will not be fully saved until you save the entire query design.
- **Close:** Close the **Filters Manager** dialog and return to the original location (Rule Manager)
- **Refresh**: Clicking **Refresh** button will refresh/reload the **Filter Manager** dialog.
- **Delete**: Clicking **Delete** button will delete the entire created filter criteria.

13.2 Filter Line Buttons

Use **()** to add a grouped filter.
Use **+** to add a filter in a group.
Use to delete a filter line.

13.3 Filter Structure

Each query filter line consists of the following components:

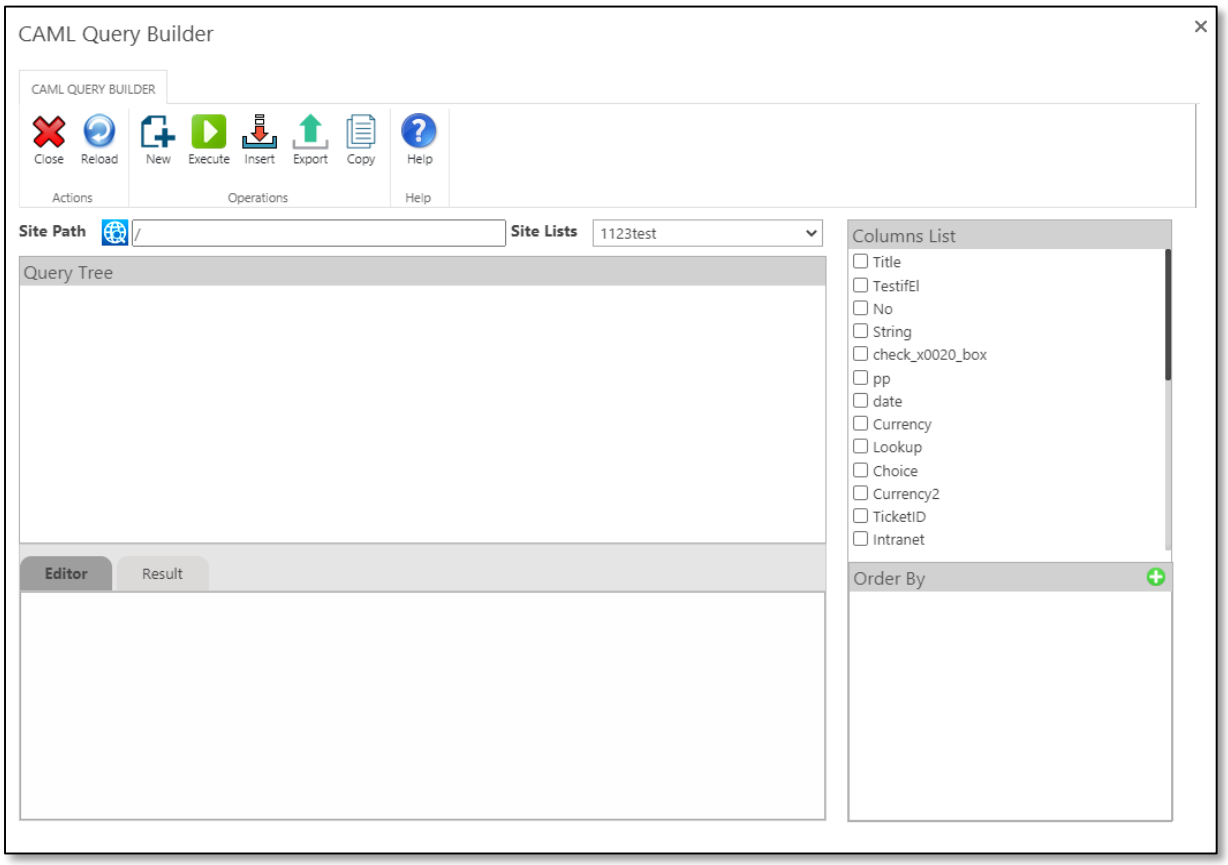
- **Entity**: Lists query's entities to select from.
- **Column**: Lists selected entity columns to select from.
- **Operator**: Lists available operators based on the selected column type.
- **Value Types**: Lists the value types:
  - **Value**: To enter a static value.
  - **Control**: Lists all available controls in the **Ad-Hoc Query Form** to select form. The filter line will retrieve the value form the selected control at

runtime, when the user click the "Run Report" button at the report's top ribbon.
    - **Column**: Lists all available columns from selected designed query's entities. The column line will retrieve the value form the selected SP entity/DB entity when the user clicks the "**Run Report**" button at the report's top ribbon at runtime.
- **Value:** This field will appear when selecting the type "Value" for the **Value Types**.

14 CAML Query Builder

The **CAML Query Builder** is an intuitive user interface that helps users/designers creating CAML Query in an easy and quick manner. It includes features to create query, export query, copy query and test the query results on the spot.

14.1 CAML Query Builder Ribbon



- **Close:** Click on this button to close the CAML Query Builder and return to the original location (Rule Manager)
- **Reload**: Click on this button to reload lists and columns. Reloading the dialog clear any unsaved data.
- **New**: Click on this button to create a new query for the selected list.
- **Execute:** Click on this button to test the query and generate the CAML Query results in the result tab.
- **Insert**: Click on this button to insert the query in the Action area in the Rule Builder.
- **Export**: Click on this button to export the CAML Query to a Text file.
- **Copy**: Click on this button to copy the generated CAML Query in the editor pane into the clipboard.

14.2 Site Path Pane

Used to specify the site you want to use the CAML Query based on.

14.3 Site Lists Pane

Used to select the list/library that you want to use in the CAML Query.

14.4 Columns List Pane

Used to select the columns that want to retrieve them. If you do not select any one, the query will retrieve all the columns.

14.5 Query Tree Pane

The area where you can build a CAML Query conditions. Each row contains four fields:

- **Column #1 (Filter / And / Or):** Used to add **AND / OR** filters, if you need to remove the **AND / OR** filter set it to type **Filter**. You can add one or more than one AND/OR conditions.
- **Column #2 (Columns' List):** Used to select the column(s) that need to filter.
- **Column #3 (CAML Query Operators):** Equal, Not Equal, Greater Than, Greater Than or Equal, Less Than, Less Than or Equal, Is Null, Is Not Null, Begins With, Contains, IN, Includes, No Includes and Date Ranges Overlap.
- **Column #4 (Field Value):** The value of field you need to filter on it.

14.6 Order By Pane

Used to add/delete "Order By" to CAML Query. You can add one or more than one Order By columns.

14.7 Editor Pane

Show the CAML Query text, it will be changed if any changes occurred on the Query Tree, Columns List and Order By.

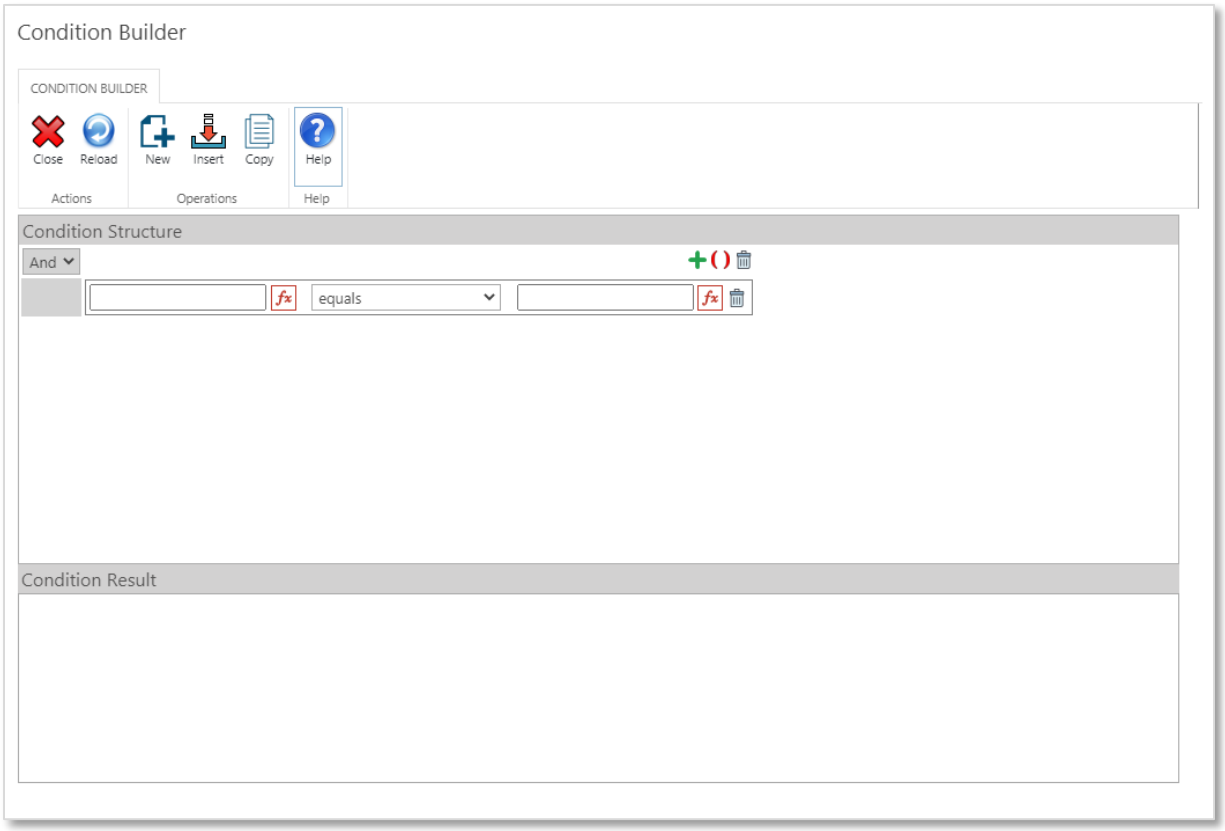
14.8 Result Pane

Show the CAML Query results to test that the generated query is working properly.

15 Conditions Builder

The **Condition Builder** is a user interface that helps users/designers building rule conditions in an easy and quick manner.

15.1 Condition Builder Ribbon



- **Close:** Click on this button to close the Condition Builder and return to the original location (Rule Manager)
- **Refresh**: Click on this button to reload the condition builder. This will empty the Condition Structure and Condition Result sections.
- **New**: Click on this button to create a new condition.
- **Insert**: Click on this button to insert the condition in the Condition area in the Rule Builder.
- **Copy**: Click on this button to copy the generated condition in the Condition Results pane into the clipboard.

15.2 Condition Structure

The area where you can build a condition.
Use **()** to add a grouped condition.
Use **+** to add a condition in a group.
Use 🗑 to delete a condition line.

15.3 Condition Result

Show the generated Condition result. It will be changed automatically if any changes occurred on the Condition Structure.

16 SPARK Reports Viewer Web Part

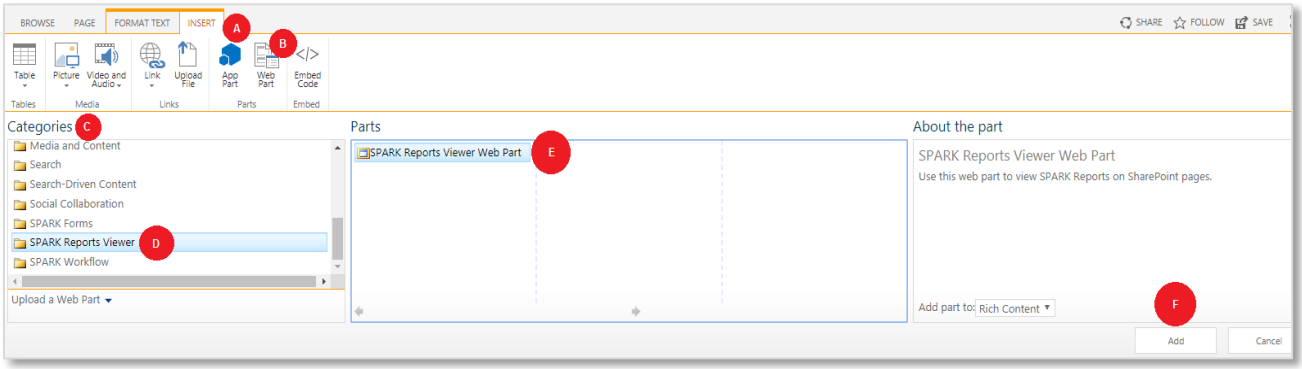
Note: This topic applies to SPARK Reports Builder Enterprise Editions only.

This topic describes how to add and configure the **SPARK Reports Viewer** web part to a SharePoint site page.

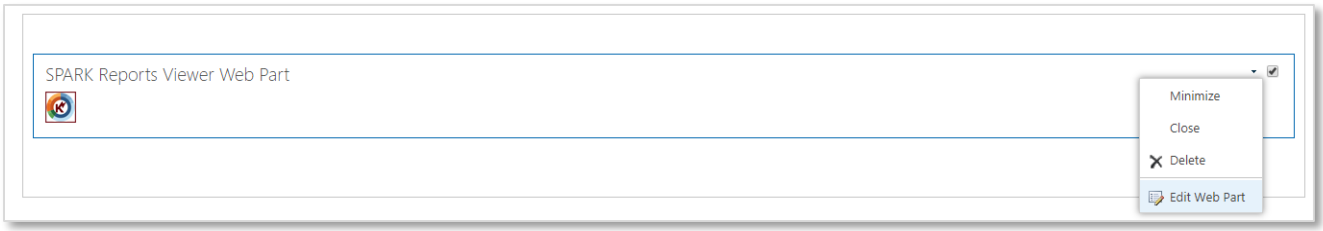
You can use **SPARK Reports Viewer Web Part** to embed SPARK reports into SharePoint pages and configure them directly there.

Here are the steps on how to add a **SPARK Reports Viewer** web part to a page:

1. Navigate to the page of the site to which you want to add the web part on.
2. Click the settings icon on the upper right and then click Edit Page.
3. Select the Insert Group, then click on Web Part, under Categories choose SPARK Reports Viewer and then click on Add.



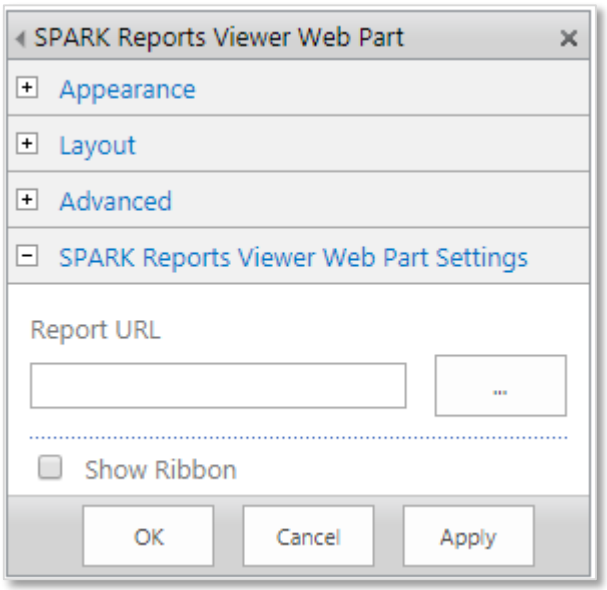
4. Click on the Edit Web Part



5. In the **SPARK Reports Viewer Web Part** area, expand the Web Part Settings and type or paste the report's URL you want to display in the web part and click OK. The page will reload the report inside the inserted web part.

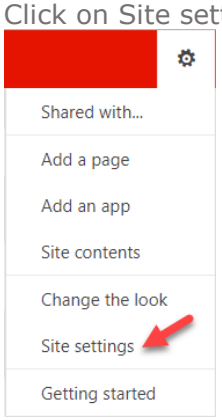
   - In order to get the report's URL, open the report from the published link, copy the URL and past it into the report's web part URL field.
   - If you want to show the form ribbon, check **Show Ribbon** property.

   You can adjust the web part setting (Appearance, Layout and Advanced) at any time as desired.
6. Under PAGE Group, click on Save to save the web part

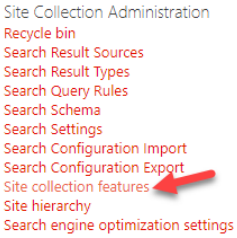


Note #1: In order to have SPARK Report's Viewer Web Part available in your site's Web Part Gallery you need to activate its feature on the site collection features first as follows:

Click on Site settings.

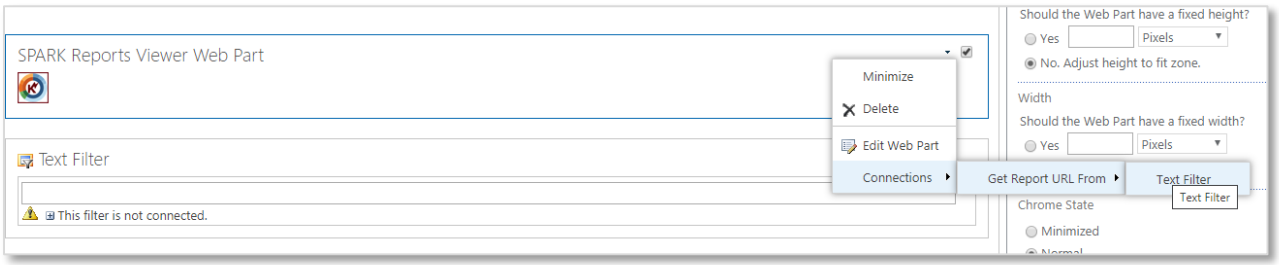


In Site Collection Administration, click on Site collection features.



Locate SPARK Reports Viewer Builder Web Part feature, then click on Activate button.

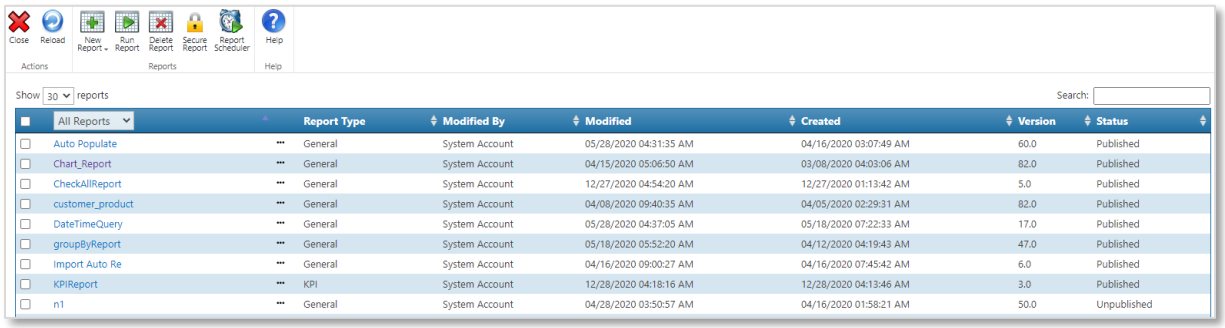
Note #2: The web part supports connecting with other web parts to dynamically passing the URL from them to the web part. For example, you can have a dropdown web part having multiple reports URLs and pass the selected URL to SPARK Reports Viewer web part to render the report dynamically and switch between reports based on user selection.



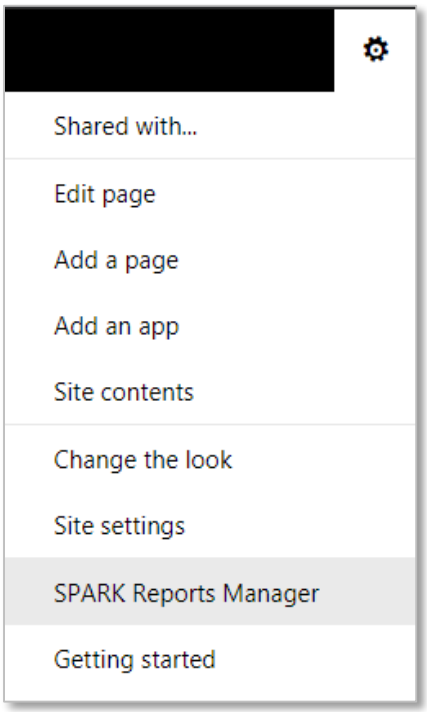
17 SPARK Reports Manager

The first page you will open when working with SPARK Reports is the **SPARK Reports Manager** page. In this page, you will be able create new reports, view, delete and manage your saved (published and unpublished) site's reports.

Note that you need to have at least "Manage lists" permission to access **SPARK Reports Manager** page.



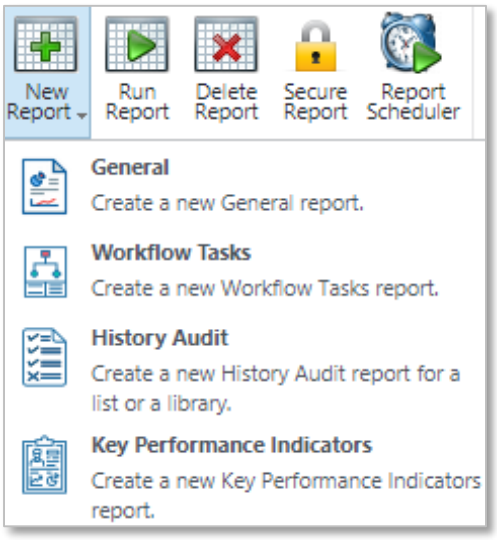
In order to access this page you need to click on **SPARK Reports Manager** on the site's top menu.



17.1 SPARK Reports Manager Ribbon

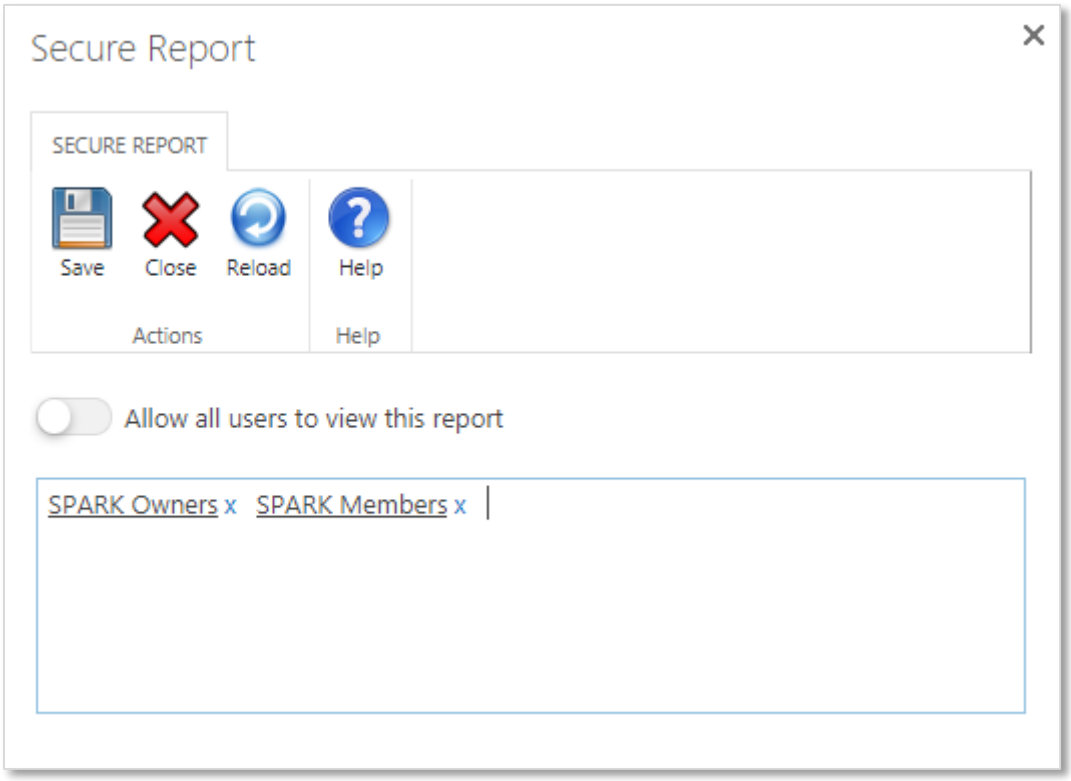
- **Close:** Click **Close** button to close **SPARK Reports Manager** page and return to the original location (Source page).

- **Reload**: Click **Reload** button to reload **SPARK Reports Manager** page.
- **New Report**: Clicking **New Report** button will open the **Report Designer Workspace** page in a "new report" mode. Refer to the **Report Designer Workspace** section for more details. This button will drilldown a submenu of four types of reports that the user can choose from:



  - **General**: This is a general type of reports, which allow you to design an unrestricted report's query and template from scratch.
  - **Workflow Tasks**: This type allows you to design workflow-tasks based reports. Specifically designed to retrieve workflow tasks lists data and represent them in specific predesigned workflow reports query and design. In this type of reports, you can create reports to show: All Users Pending Tasks, All Users Tasks, All Users Complete Tasks, Me and My Employees Pending Tasks (Organization Hierarchy), only a specific Group Members Tasks and Only My "Current Logged in User" Pending Tasks.
  - **History Audit**: This type allows you to design an audit report for all changes and modifies that occurred on a specific list or library, you will find all related version history data in the report grouped by the modified user name, date and time of modifying.
  - **Key Performance Indicator**: This type allows you to design a KPI reports showing the performance of the organization staff/users based on their workflow assigned tasks following up and completion times.
    In this type of reports you can create reports to show the KPIs for All Users Pending Tasks, All Users Tasks, All Users Completed Tasks, Me and My Employees Pending Tasks (Organization Hierarchy), Only a specific Group Members Tasks and Only My "Current Logged in User" Pending Tasks.

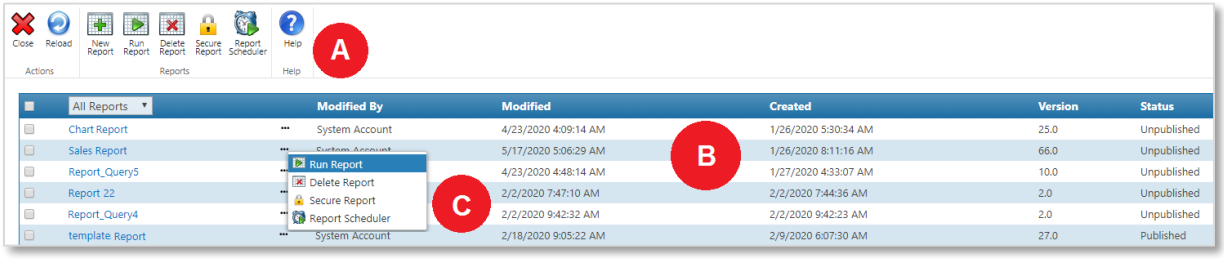
- **Run Report**: Click on this button to run the selected report. This button will open a new tab in your browser to run the report within, so you need to make sure you allow opening tabs on your browser. You also need to select only one published report before clicking this button.
- **Delete Report**: Click on this button to delete selected report(s). A confirmation dialog will appear before deleting proceeds.
- **Secure Report**: Click on this button to secure the selected report, and prevent users from accessing it. The default security settings is to allow all users to view/run it. This button will open a security management dialog for the selected report. You also need to select only one report before clicking this button.



  - **Allow all users to view this report**: Switching this option **ON** will enable all users who have access to the report query's data sources to view and run the report.
  - **Access list**: This section is a people and groups input field to all the report own/administrator to select users and groups who will have access to the report and will be able to run it. This field will be hidden in case the **Allow all users to view this report** option is switched **ON**.

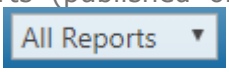
- **Report Scheduler**: Click on this button to create a scheduler for the selected report, to run it at specified time(s). This button will open the Report Scheduler management dialog for the selected report. You also need to select only one report before clicking this button. Refer to **Report Scheduler** section for more information.

17.2 SPARK Reports Manager Structure



SPARK Reports Manager page's structure consists of the following parts:

A- **Top Ribbon:** refer to **SPARK Reports Manager Top Ribbon** Section for more details.

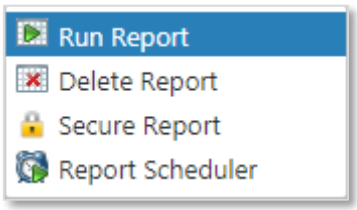
B- **Reports List:** In this list, you will find all your saved reports (published or unpublished) ones. You can filter this list by using the filter box 

.

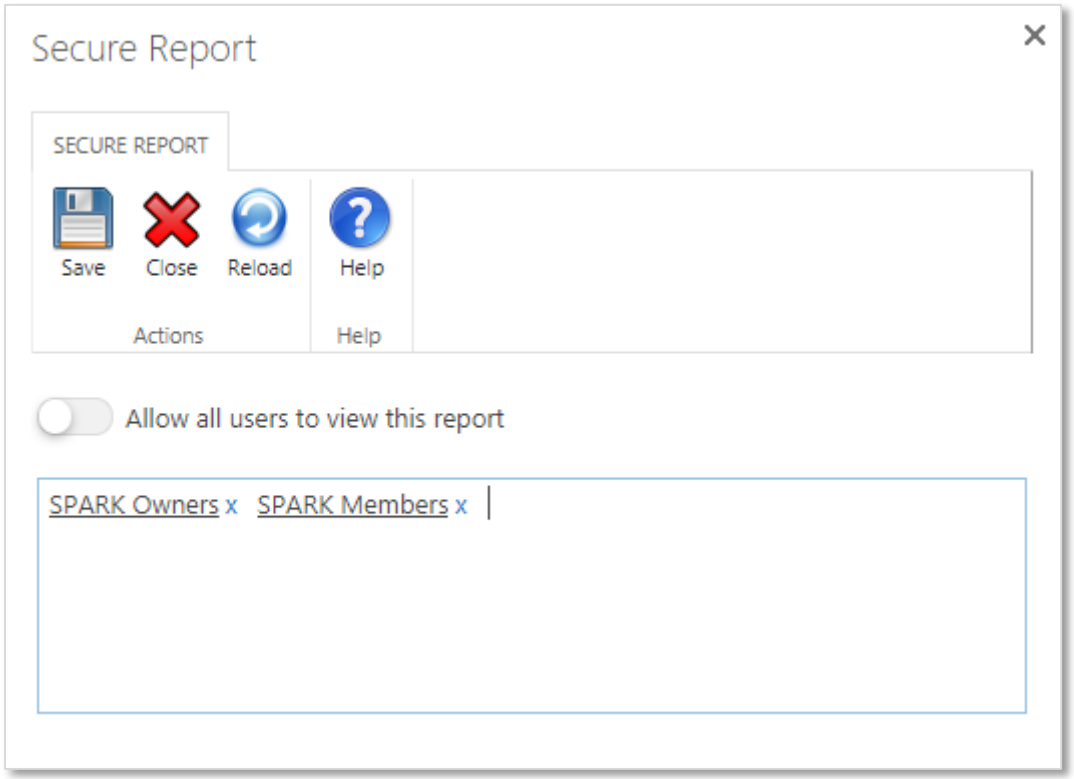
In addition, you can select all reports by checking the upper list's checkbox  . The lists consists of the following report details (columns):

a. **Report Name:** Displays the name of the report.
b. **Report Type:** Displays the type of the created report [General, Workflow Tasks, History Audit and Key Performance Indicator].
c. **Modified By:** Displays the name of the last user who modified the report.
d. **Modified:** Displays the last date and time this report was modified on.
e. **Created:** Displays the date and time this report was created on.
f. **Version:** Displays the current version number of the report.
g. **Status:** Displays the status of the report.

C- Report Menu: Click on the three dots next to the report's name to display the **Report Menu**, which consists of the following functions:



- **Run Report**: Click on this button to run the report. This button will open a new tab in your browser to run the report within, so you need to make sure you allow opening tabs on your browser. This button will be disabled if the report is unpublished.
- **Delete Report**: Click on this button to delete the report. A confirmation dialog will appear before deleting proceeds.
- **Secure Report**: Click on this button to secure the report, and prevent users from accessing it. The default security settings is to allow all users to view/run it. This button will open a security management dialog for the report.



- **Allow all users to view this report**: Switching this option **ON** will enable all users who have access to the report query's data sources to view and run the report.
- **Access list**: This section is a people and groups input field to all the report own/administrator to select users and groups who will have access to the report and will be able to run it. This field will be hidden in case the **Allow all users to view this report** option is switched **ON**.

- **Report Scheduler**: Click on this button to create a scheduler for the report, to run it at specified time(s). This button will open the Report Scheduler management dialog for the report. Refer to **Report Scheduler** section for more information.

17.3 SPARK Report Scheduler

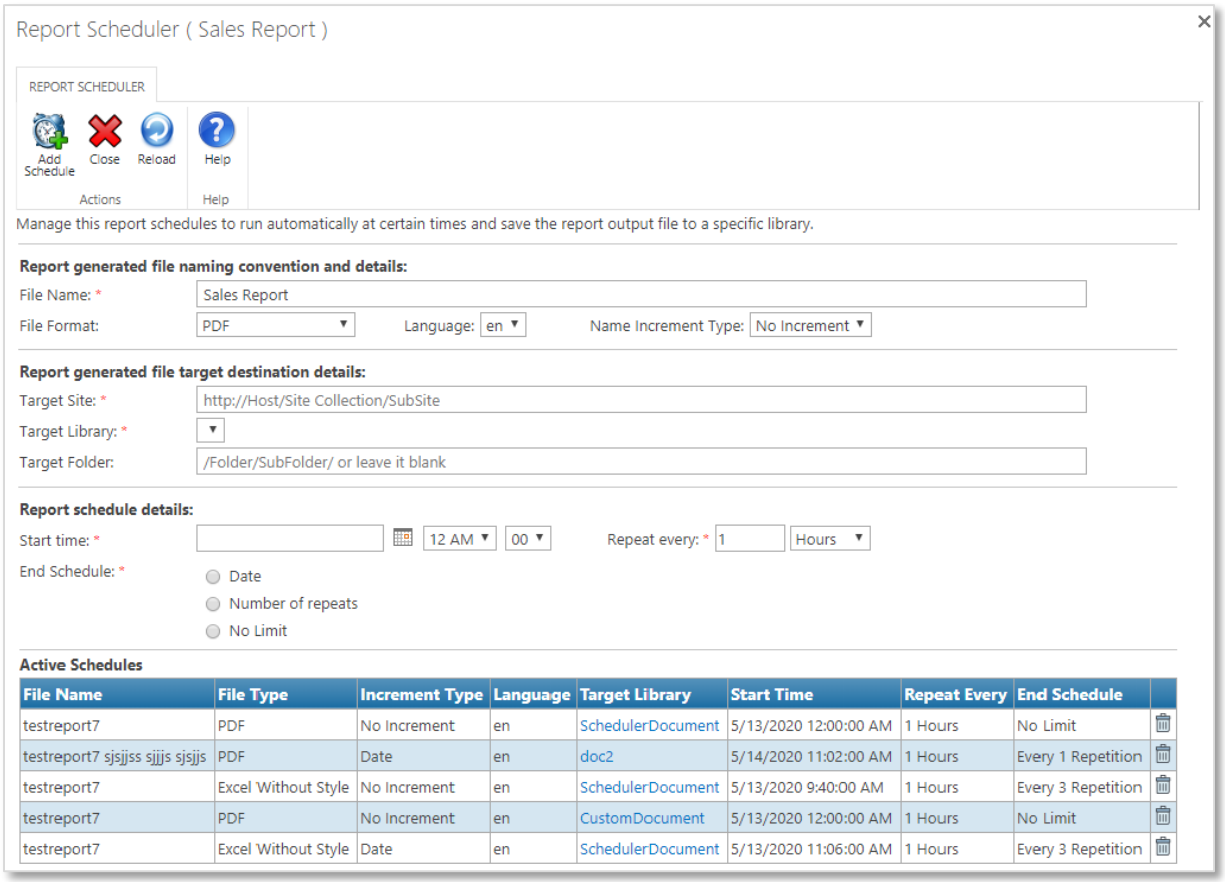
Note: This topic applies to SPARK Reports Builder Enterprise Edition only.

SPARK Reports Builder enable users and designers to schedule the running process of reports automatically at specific dates and times. The scheduler engine, which depends on a SharePoint Timer job, will save the output of the generated report to SharePoint document library specified in the report's scheduler settings. The user/designer will be able to create multiple schedulers for a single report and saves their output files to different locations on the SharePoint farm site's libraries.

Note that **SAPRK Reports Scheduler Service** must be configured in the Central Admin web application in order to be able to use this feature. Refer to the **Installation Guide** for more information about how to configure and activate this feature on the central administration web application.

You can configure a schedule for a report as follows:

- Click on the **Site Settings** icon, and then click on **SPARK Reports Manager** link.
- Select the report then click **Report Scheduler** button at the top ribbon, or click at the three dots next to the reports name then click on Report Scheduler button from the report's menu.



- On **SPARK Report Scheduler** manager page do the following:
  - To add a report schedule, specify the following attributes then click on the **Add Schedule** button at the top ribbon:
    - **File Name:** Set the report's output file name. By default, the name will be the same as the report's name and you can customize it the way you want.
    - **File Format:** Select the file format you want to save the report's output file with, when the scheduler runs it at the specified date/time. You can choose form the following file formats [PDF, Word, Excel without style and Excel with style].
    - **Report Language:** Select the report language you want to schedule if you have designed multilingual report. If not; then there will be one selected language in the field.
    - **Name Increment Type:** Specify how the report's output file would be handled when the scheduler run it:
      1- **No Increment:** No changes will be done on the file name and it will be saved over the same file name if exists in the library. If the versions are enabled on the library then you will have a new version of the output file each time the scheduler runs.

2- **Count:** Will add an increment number to the report's output file name each time the scheduler runs. This option will prevent file name duplicate when saving to the library.
3- **Date Time:** Will add current Year_ Month_ Day_ Hour_ Minute_ Second combined string to the report's output file name each time the scheduler runs. This option will prevent file name duplicate when saving to the library.

- **Target Site:** Specify the target site's URL of the library that you want to save the report's output file into it.
- **Target Library:** Specify the target library's name that you want to save the report's output file into it. This list will be populated with site's libraries once you enter the site's Path URL.
- **Target Folder (optional):** Specify the target library's folder that you want to save the report's output file into it. This is an optional field and you can leave it blank if you want to save to the root folder of the **Target Library**.
- **Start time:** Specify the date and time to run the report's schedule on.
- **Repeat every:** Specify the frequency for running the report in hours, days or months. This field accepts only numeric values.
- **End Schedule:** Specify the end of the report's schedule, either by:
  - Specifying a specific **End Date/Time**
  - Specifying the **Number of Repeats**
  - Setting the report's schedule to run with **No Limit** (indefinitely).

- To delete an existing report's schedule, click on the **Delete** icon in the registered schedules table.

18 Naming conventions

Naming conventions is important for the following:

1. It makes it easy to know what the control type is.
2. It is consistent and allows any team member working on a report design to easily understanding the logic of the report and the Ad-Hoc query form.
3. It makes it easier to come up with new names for controls.
4. There are less errors, which makes it easy to distinguish one control from another if there are many controls.

The naming conventions of the control always describe the content it stores. The conventions are one word and written with camel case capitalization style. With camel case, the first letter is lower case and the first letter of each subsequent concatenated word is capitalized. We also prefix the control by a shortening the first two to four characters of the type of control it is. For example, "txtProjectCode" tells us we are storing project code in a single line of text field.

Here are prefixes we use with controls:

Category	Control	Prefix
General	Label	lbl
	Horizontal Line	hl
	Vertical Line	vl
	HyperLink	hl
	Page Viewer	pv
	Image	img
Input	Button	btn
	TextBox (single line of text)	txt
		num (when the control is Number Only)
	TextArea (multiple lines of text)	mtxt
	Ritch Text Editor	rtxt
	CheckBoxList	cbl
	CheckBox	cb
	DropDownList	ddl
	Date	dat
	Time	tim
	DateTime	date
	RadioButton	rb
	Toggle Switch	ts
	Panel	pnl
	Tab	tab
Integration	Lookup	lok
	Advanced Lookup	alok
	People Picker	Pp
	External Data Dialog	edd
	SQL Connector	sql
	XML Connector	xml
	Web Connector	web
	Electronic Signature	sig
Advanced	Barcode	bc
	QR	Qr

Here are prefixes we use with other objects:

Object	Prefix
Footer Cell	ftcell
Calculated Column	calcol
Rule	rul
Report Page	rpg
Function	fn