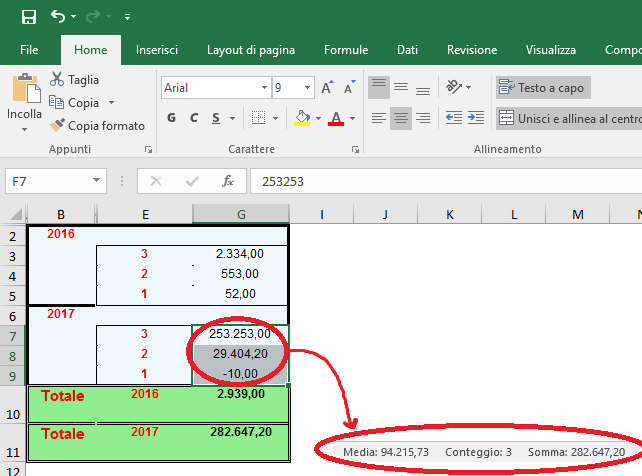
### PERCHE’ USARE REPORTING SERVICES

Usando i reporting services ho notato alcune funzionalità molto utili:

* Funzionalità interattive completamente gratuite (drill down, drill through, interactive sorting);
* Integrati in visual studio;
* Possono essere visualizzati sia su pagina web (server report) che su windows form (server e local report);
* Possono essere esportati in molti formati: (png, excel, word, csv, pdf, xml, html);
* Si possono usare dei template grafici (creati con paint, photoshop, …) per semplificare e redere più veloce la creazione del report;
* Più veloci dei report con Crystal (ho misurato il tempo di esecuzione del report “bilancio consuntivo” -dal momento in cui ho premuto il bottone per visualizzare il report, fino a quando il report è stato visualizzato- ed effettivamente ci ha messo circa il 50% del tempo in meno)

Altri vantaggi/funzionalità scoperte durante questo studio sono state:

* La possibilità di creare i report dinamicamente a runtime da C#
* Si possono passare i parametri di input dai report (funzionalità presente anche in Crystal)
* Ai local report è possibile passare da C# anche il datasource, il (i) dataset (in realtà IL DataTable)
* I report sono scritti in XML
* L’estensione dei server report è .rdl mentre dei local report .rdlc
* Molti items: Casella di testo, lista, tabella, matrice, rettangolo, elenco, immagine, sottoreport, grafico, misuratore, mappa (anche se solo Americana), barra dei dati, grafico sparkline, indicatore
* L’esportazione in Excel è particolarmente efficiente in quanto il report rimane ben formattato



### LOCAL REPORT VS SERVER REPORT

There was a similar discussion in this group previously. You can refer the same: http://social.msdn.microsoft.com/Forums/en/vsreportcontrols/thread/1b06b84a-3c6e-40f1-908e-07e332a45a04 [ReportViewer control local and remote mode differences]

Both Local Mode and Remote mode has their own advantages.. This is completely based on your requirement you can choose Local mode or remote mode of report processing.

When to Use Local Processing

Local processing mode is recommended for applications that include small to medium-sized reports and datasets. Because all data and report processing is performed on the client, you might encounter performance degradation or memory limitations if you attempt to process large or complex reports and queries.

Local processing mode is less scalable than remote processing and is intended for standalone applications that do not require a report server. Users who are familiar with server reports that run on a remote SQL Server Reporting Services report server should note the following differences:

In remote processing mode, the report server both retrieves the data and processes the report, whereas in local processing mode, the ReportViewer control processes the report only, and the the host application must retrieve the data and supply it to the ReportViewer control.

Client report definitions do not include query information.

Report parameters in client report definitions (.rdlc) do not map to data source query parameters.

There is no parameter input area for a client report definition that accepts values that are subsequently used in a query

When to Use Remote Processing

Server-based reporting lets you centralize report storage and management, set policies and secure access to reports and folders, control how reports are processed and distributed, and standardize how reports are used in your business. Reporting Services can be installed in single-server, distributed, and clustered configurations. Consider using remote processing if your report has the following characteristics:

Report will be accessed by many users.

Report has a very complex query or contains a very large amount of data, resulting in system resource overload on the computer that hosts the application.

Report is already published on a report server, and you want to include it an application that you create and deploy.

Reference articles:

http://msdn.microsoft.com/en-us/library/ms251704.aspx [Configuring ReportViewer for Local Processing]

http://msdn.microsoft.com/en-us/library/ms252075(VS.80).aspx [Configuring ReportViewer for Remote Processing]

In generale nel nostro caso specifico l'uso di un server report credo possa essere dannoso in quanto nel live update bisogna includere un file che aggiorni il db con il nuovo report, mentre con i local report si mette in live il report aggiornato e stop senza aggiornare il db.

Server report e local report si creano entrambi dal tool grafico quindi a livello di complessità di creazione di report siamo sullo stesso livello. Nei local report però da codice possiamo passare dataset, datasource (oltre ai parametri di input che si possono passare anche nei server report).

In generale ho notato che un local report si presta di più per essere visualizzato nel windows form rispetto al server report che è quasi come fosse progettato per la versione web.

### REPORT DA C#

Le classi che sto scrivendo in c# sui report permettono di creare report da codice dinamicamente.

Il limite più grande è causato dal fatto che bisogna espandere le classi per le varie funzionalità/item da implementare e codificare nel formato XML (formato standard dei report)

Ovviamente i report fatti da codice sono "più complessi" da fare rispetto a quelli fatti da tool grafico in quanto il risultato lo si può vedere solo quando si manda in run il progetto.

I vantaggi sono molteplici:

+report dinamico;

+creato a run time;

+si potrebbe implementare in easy la possibilità di far fare ai clienti report custom tramite qualche form o tool che aiuti graficamente senza che scrivano codice.

### Creare local report e visualizzarli in un form windows

* Usare il template MS “applicazione report”
* Creare un dataset
* Creare e modificare il report (Report.rdlc)
* Nel Form.Designer.cs è importante che ci sia il report viewer per visualizzare il report nel form e il form usi il metodo load.

//

// reportViewer1

//

this.reportViewer1.Anchor = ((System.Windows.Forms.AnchorStyles)((((System.Windows.Forms.AnchorStyles.Top | System.Windows.Forms.AnchorStyles.Bottom)

| System.Windows.Forms.AnchorStyles.Left)

| System.Windows.Forms.AnchorStyles.Right)));

this.reportViewer1.Location = new System.Drawing.Point(1, 3);

this.reportViewer1.Name = "reportViewer1";

this.reportViewer1.Size = new System.Drawing.Size(1044, 615);

this.reportViewer1.TabIndex = 0;

//

// Form1

//

....

this.Load += new System.EventHandler(this.Form1\_Load);

Ora in Form.cs scrivere il metodo Form\_Load

private void Form1\_Load(object sender, EventArgs e) {

LocalReport localReport = reportViewer1.LocalReport;

reportViewer1.Reset();

reportViewer1.ProcessingMode = ProcessingMode.Local;

reportViewer1.LocalReport.ReportPath = AppDomain.CurrentDomain.BaseDirectory + "DataS.rdlc";

localReport.ReportPath = AppDomain.CurrentDomain.BaseDirectory + "DataS.rdlc";

VistaForm DS = new VistaForm();

DataTable a = DS.mandate;

DataTable b = DS.mandatedetail;

a.TableName = "mandate";

string connString =

"Data Source=serversw12;" +

"Initial Catalog=unirc\_easy;" +

"User id=assistenza;" +

"Password=afrodite;";

//@"your connection string here";

string query = "select \* from amministrazione.mandate";

SqlConnection conn = new SqlConnection(connString);

SqlCommand cmd = new SqlCommand(query, conn);

conn.Open();

// create data adapter

SqlDataAdapter da = new SqlDataAdapter(cmd);

// this will query your database and return the result to your datatable

da.Fill(a);

conn.Close();

da.Dispose();

var RDS = new ReportDataSource("DataSet1", a);

reportViewer1.LocalReport.DataSources.Add(RDS);

// Refresh the report

reportViewer1.RefreshReport();

}

### Visualizzare server report in un form windows

* Usare il template MS “windows form application”
* Nel Form.Designer.cs è importante che ci sia il report viewer per visualizzare il report nel form e il form usi il metodo load.

//

// reportViewer1

//

this.reportViewer1.Anchor = ((System.Windows.Forms.AnchorStyles)((((System.Windows.Forms.AnchorStyles.Top | System.Windows.Forms.AnchorStyles.Bottom)

| System.Windows.Forms.AnchorStyles.Left)

| System.Windows.Forms.AnchorStyles.Right)));

this.reportViewer1.Location = new System.Drawing.Point(1, 3);

this.reportViewer1.Name = "reportViewer1";

this.reportViewer1.Size = new System.Drawing.Size(1044, 615);

this.reportViewer1.TabIndex = 0;

//

// Form1

//

....

this.Load += new System.EventHandler(this.Form1\_Load);

Ora in Form.cs scrivere il metodo Form\_Load

private void Form1\_Load(object sender, EventArgs e) {

reportViewer1.ProcessingMode = ProcessingMode.Remote;

ServerReport serverReport = reportViewer1.ServerReport;

// Get a reference to the default credentials

System.Net.ICredentials credentials =

new System.Net.NetworkCredential("administrator", "Juswafrigi.33893");

// Get a reference to the report server credentials

ReportServerCredentials rsCredentials =

serverReport.ReportServerCredentials;

// Set the credentials for the server report

rsCredentials.NetworkCredentials = credentials;

// Set the report server URL and report path

serverReport.ReportServerUrl =

new Uri("http://ufficio1.temposrl.it:8085/ReportServer");

serverReport.ReportPath =

"/Verifica saldo/Report1";

string[] stringarr = new string[] { "ayear", "2017", "date", "20/06/2017" };

for (int i = 0; i < stringarr.Length; i++) {

ReportParameter param = new ReportParameter();

param.Name = stringarr[i];

i++;

param.Values.Add(stringarr[i]);

reportViewer1.ServerReport.SetParameters(new ReportParameter[] {param});

}

reportViewer1.ShowParameterPrompts= false;

reportViewer1.Name = "Verifica Saldo";

// Refresh the report

reportViewer1.RefreshReport();

### Personalizzare il ReportViewer e alcune info

* Digitando reportViewer1. è possibile accedere ai parametri di modifica (es zoom, ShowParameterPrompts, ecc)
* ServerReport serverReport = reportViewer1.ServerReport;

reportViewer1.ProcessingMode = ProcessingMode.Remote; SE SERVER REPORT

LocalReport localReport = reportViewer1.LocalReport;

reportViewer1.ProcessingMode = ProcessingMode.Local; SE LOCAL REPORT

* PathServer report

Se si copia il link che ci viene fornito dalla pagina web non funziona correttamente

// Set the report server URL and report path

serverReport.ReportServerUrl =

new Uri("http://ufficio1.temposrl.it:8085/ReportServer");

serverReport.ReportPath =

“/Pages/ReportViewer.aspx?%2fVerifica+saldo%2fReport1&rs:Command=Render ";

(<http://serversw12/ReportServer/Pages/ReportViewer.aspx?%2fVerifica+saldo%2fReport1&rs:Command=Render>)

È necessario pertanto fornire il path in questa maniera:

// Set the report server URL and report path

serverReport.ReportServerUrl =

new Uri("http://ufficio1.temposrl.it:8085/ReportServer");

serverReport.ReportPath =

"/Verifica saldo/Report1";



* Ai local report è necessario assegnare un ReportDataSource al dataset altrimenti andrà in errore:

var RDS = new ReportDataSource("DataSet1", a);

reportViewer1.LocalReport.DataSources.Add(RDS);

Ovviamente il nome del dataset (“DataSet1”) deve essere il nome che abbiamo dato al dataset nel Report.rdlc

### Convertire .RDL e .RDLC

<https://msdn.microsoft.com/en-us/library/ms252109(v=vs.120).aspx>

**ATTENZIONE questa procedura serve solo per convertire da .rdl a .rdlc e vice versa ma a patto che la versione di schema XML non cambi!!**

**(es. se partiamo da un file .rdl con schema 2003, con la conversione otterremo un file .rdlc 2003 e non 2010 o 2016)**

**Per maggiori informazione sugli schemi consulare il capitolo relativo (**[Report Schema](#_Report_Schema)**)**

Bisogna partire fancendo un punto della situazione sulle differenze tra i due formati:

* Sia il report .rdl (server) che .rdlc (local) sono scritti in XML.
* Usano gli stessi tipi di item.
* The <Query> element of RDL contains query or command information and is used by the report server to connect to the data sources of the report.
* The <Query> element is optional in an RDLC file. This element is ignored by the **ReportViewer** control because the **ReportViewer** control does not perform any data processing in local processing mode, but uses data that the host application supplies.

**Da .rdlc a .rdl**

Converting an .rdlc file to the .rdl format is automatic if the following conditions are true:

* Report data is provided through a data table.
* The data source used by the .rdlc file maps to one of the data processing extensions that is installed on the report server (for example, if the data source is a SQL Server database, you can use the SQL Server data processing extension that is installed by default on the report server).
* The query and data source connection information do not need to be modified.

To convert the .rdlc file automatically, rename the file name extension to .rdl, and then publish or upload the .rdl file to a report server.

To manually convert an .rdlc to an .rdl file format, follow these steps:

1. Determine the data source that you want to use with the .rdl file. You can use a SQL Server relational database, an Analysis Service multidimensional database, an Oracle database, or any of the other supported data sources. The schema must be the same as the one you used to create the original .rdlc report layout.
2. Rename the .rdlc file to use the .rdl file extension.
3. In SQL Server, in Business Intelligence Development Studio, create or open the Report Server project that will contain the report.
4. On the **Project** menu, click **Add Existing Item**. Select the .rdl file you created in the first step and add it to the project.
5. In Solution Explorer, open the .rdl file you just added to the project. The report definition opens in design mode.
6. In the **View** menu, select **Report Data**.
7. In the **Report Data** window, select the first data source and then click the **Edit** button.
8. In the **Data Source Properties** dialog box, click the **Edit** button by the data source name.
9. Update the connection information. You should replace the connection string so that the report connects to the data source you want to use. You might need to update the data source type, server, authentication options, and database selection. When finished, click **OK** to close the dialog box.
10. In the **Report Data** window, select the first dataset associated with the data source you edited, and then click the **Edit** button.
11. In the **Dataset Properties** dialog box, click **Query Designer**.
12. Click **Run** to verify that the query string returns the data you expect.
13. Repeat the previous steps to modify any other dataset and data sources used in the report.
14. Preview the report to verify that the data is merged into the report layout as expected. To preview the report, click the **Preview** tab.

**Da .rdl a .rdlc**

When converting a .rdl file to .rdlc format, you must manually replace the data source and query information in the report definition with data constructs provided in your application.

1. Rename the .rdl file to use the .rdlc file extension.
2. In Visual Studio, open the solution or project that will contain the report.
3. Create or add the dataset that defines the data you want to use to your projector solution. For more information, see [Creating Data Sources for a ReportViewer Report](https://msdn.microsoft.com/en-us/library/ms252094(v=vs.80).aspx).
4. On the Project menu, click **Add Existing Item** and then select the .rdlc file that you created in the first step.
5. In the project, open the form or Web page that will contain the report.
6. From the Toolbox, in the Data group, drag a ReportViewer control onto the form or Web page.
7. In the **ReportViewer Tasks** smart tags panel, in **Choose Reports**, select the .rdlc file to use with the control.
8. In the **ReportViewer Tasks** smart tags panel, in **Choose Data Sources**, select the dataset you want to use. The dataset object, binding source object, and table object will appear at the bottom of the workspace. For more information about updating data source references, see [Updating and Rebinding Data Source References](https://msdn.microsoft.com/en-us/library/ms252085(v=vs.80).aspx).
9. Save all files, and then build or deploy the project to verify that the report contains the data you expect.

### Creare REPORT C# a runtime

Essendo il report scritto in XML l’idea è che si potrebbe scrivere in C# e convertire il codice in XML tramite System.Xml.Linq;

Mi sono aiutato molto da questo link che mi ha passato Maria e da dove ho preso varie delle classi presenti in Report -> Rdl

<https://www.codeproject.com/articles/705248/generating-rdlc-dynamically-for-the-report-viewer>

Inoltre

<https://technet.microsoft.com/en-us/library/ms170667(v=sql.105).aspx>

e

<https://msdn.microsoft.com/en-us/library/ee240083(v=sql.105).aspx>

### TIPI ACCETTATI DAL REPORT

* String (UNICODE)
* Integer > - 2147483649 e < 2147483648
* Boolean
* Float (32 bit)
* DateTime (date and time-of-day)
* [Expression](#_EXPRESSION_SYNTAX)
* RdlSize: Integer non negativo seguito da: pt (points) o pc (picas) o in (inches) o mm (millimeters) o cm (centimeters)
* Colori: Passarli come System.Drawing.Color.X.Name

| **Tipi CLR** | **Tipo di dati corrispondente** |
| --- | --- |
| Boolean | Boolean |
| DateTime, DateTimeOffset | DateTime |
| Int16, Int32, UInt16, Byte, SByte | Integer |
| Single, Double | Float |
| String, Char, GUID, Timespan | String |

### Glossario

* **Data element**: An element that describes how [report](https://msdn.microsoft.com/en-us/library/ee240603(v=sql.105).aspx#gt_556439b8-0249-44d1-894c-6c7dbd8f0a00) data is to be represented in a text rendering.
* **Data point**: A prototype for the data values that are displayed by a chart series. Data points can be displayed in different shapes depending on the [chart](https://msdn.microsoft.com/en-us/library/ee240603(v=sql.105).aspx#gt_8e07039d-d1d3-4336-a478-f35e8cacc26c) type.
* **Data provider**: A known data source that is specific to a target type and that provides data to a collector type.
* **Data rendering**: A rendering of a [report](https://msdn.microsoft.com/en-us/library/ee240603(v=sql.105).aspx#gt_556439b8-0249-44d1-894c-6c7dbd8f0a00) that is intended for data processing by a machine and that does not necessarily contain any visual information for human reading.

### EXPRESSION SYNTAX

Spesso sarà possibile inserire espressioni, quindi è opportuno specificare la sintassi utilizzata da queste:

“In addition to supporting constants as element values, RDL supports an expression language. All expressions in RDL MUST begin with the character "=" and MUST be specified in a Visual Basic–compatible syntax.

Expression values that do not begin with "=" are treated as constants of the type expected by the property if that type is [Boolean](https://msdn.microsoft.com/en-us/library/ee239929(v=sql.105).aspx), [String](https://msdn.microsoft.com/en-us/library/ee223804(v=sql.105).aspx), or [Integer](https://msdn.microsoft.com/en-us/library/ee208948(v=sql.105).aspx).”

Per inserire i valori presenti nel dataset: Parameters!NOMECAMPO.Value

Se necessario specificare anche il dataset (Parameters!NOMECAMPO.Value, ”NOME DATASET”)

Per le stringhe “Stringa”

Per commentare usare ‘riga commentata

Più avanti nella guida si troveranno tag come:

  <Bookmark>String</Bookmark>

  <Top>RdlSize</Top>

  <TAG>TipoX</TAG>

Tutti questi possono essere trasformati in espressioni con le giuste precauzioni ossia:

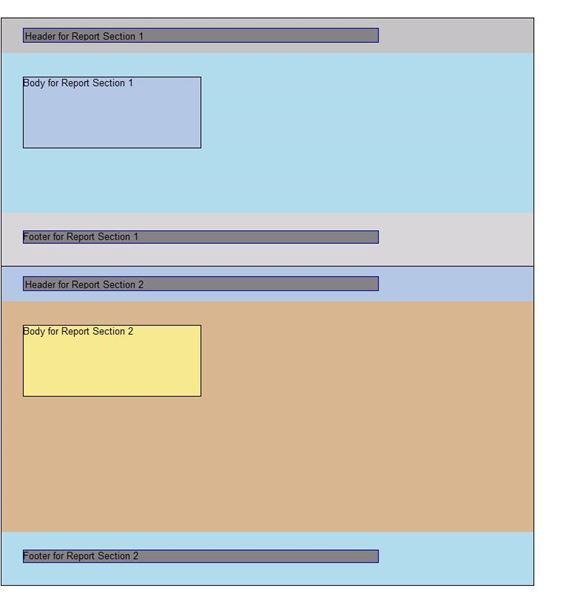
* Il tipo deve rimanere quello corretto! Se abbiamo ad esempio <TAG>Float</TAG> è necessario che l’espressione ci restituisca un float;
* Se serve un boolean si può usare ad esempio IIF (condizione, true, false);
* Se servono stringhe si potrebbe fare un X.ToString();
* Devono essere rispettate eventuali condizioni di range, es. Intenger > N;
* Devono esere rispettati condizioni sulle stringhe, es. accettato solo valori come “Dotted, Solid,…”

### ReportItems

“The **ReportItems** element specifies the collection of [report items](https://msdn.microsoft.com/en-us/library/ee240603(v=sql.105).aspx#gt_c6f8e999-fca9-4e79-96e7-fb4c2c43d601) used to define the contents of a region of a [Report](https://msdn.microsoft.com/en-us/library/ee224052(v=sql.105).aspx)”.

|  |  |
| --- | --- |
| Matrice (Tablix) | Elemento opzionale che può essere inserito più di una volta.  È un grid layout formato da righe e colonne che possono essere (in modo opzionale) filtrate, ordinate, raggruppate, ripetute e nidificate  La tablix non può essere inserita nell’header e/o nel footer. |
| Casella di testo (Textbox) | Elemento opzionale che può essere inserito più di una volta.  Può essere inserito nell’header e/o nel footer. |
| Rettangolo (Rectangle) | Elemento opzionale che può essere usato come contenitore |
| Subreport | Elemento opzionale. |
| Immagine (Image) | Elemento opzionale che può essere inserito più di una volta.  Visualizza una immagine. |
| Grafico (Chart) | Elemento opzionale che può essere inserito più di una volta.  Visualizza un grafico che aiuta a riassumere i dati in formato visivo |
| Linea (Line) | Elemento opzionale che può essere inserito più di una volta.  Visualizza una linea. |

### Struttura base del report

[](https://msdn.microsoft.com/en-us/library/ee209360(v=sql.105).aspx)

Il report ha una struttura XML e l’esempio seguente è una struttura tipo di un report:

<?xml version="1.0" encoding="utf-8"?>

<Report xmlns="http://schemas.microsoft.com/sqlserver/reporting/2010/01/reportdefinition">

<DataSources>

…

</DataSources>

<DataSets>

…

</DataSets>

<ReportSections>

<ReportSection>

<Body>

<ReportItems>

</ReportItems>

</Body>

<Page> (che contiene Header e Footer della sezione)

</Page>

</ReportSection>

</ReportSections>

</Report>

### Report Schema

Lo schema utilizzato descrive la struttura del report (Report Definition Language).

**!!ATTENZIONE!!** RDL e RDLC usano uno schema identico tranne per la parte relativa ai [Datasources](#_DataSources) e [Datasets](#_Datasets)

Report Definition Language (RDL) è una rappresentazione XML di una definizione di un report di SQL Server Reporting Services. Una definizione del report contiene informazioni sul layout e sul recupero dei dati per un report. RDL è costituito da elementi XML che corrispondono a una grammatica XML creata per Reporting Services. È possibile aggiungere funzioni personalizzate per il controllo dei valori degli elementi del report, degli stili e della formattazione mediante l'accesso agli assembly di codice all'interno dei file di definizione dei report.

RDL promuove l'interoperabilità di prodotti per la creazione di report commerciali mediante la definizione di un schema comune che consente lo scambio di definizioni dei report. Con RDL è possibile utilizzare qualsiasi protocollo o interfaccia programmatica appropriato per XML. RDL consiste in:

* XML Schema per le definizioni dei report.
* Un formato di interscambio per aziende e terze parti.
* Uno schema estensibile e aperto che supporta spazi dei nomi aggiuntivi ed elementi personalizzati.

Uno dei modi più semplici per generare codice RDL da un'applicazione consiste nell'utilizzare il Microsoft .NET Framework classi del [System.Xml](https://msdn.microsoft.com/library/system.xml(v=SQL.130).aspx)dello spazio dei nomi e [System.Linq](https://msdn.microsoft.com/library/system.linq(v=SQL.130).aspx) dello spazio dei nomi. In particolare, la classe XmlTextWriter può essere usata per scrivere codice RDL. Con XmlTextWriterè possibile generare una definizione di report completa in qualsiasi applicazione .NET Framework. Gli sviluppatori possono inoltre estendere il linguaggio RDL aggiungendo elementi del report personalizzati con proprietà personalizzate.

<Report xmlns="Link">

Dove Link è =

<http://schemas.microsoft.com/sqlserver/reporting/2010/01/reportdefinition/>

**or**

<http://schemas.microsoft.com/sqlserver/reporting/2008/01/reportdefinition/>

**or**

<http://schemas.microsoft.com/sqlserver/reporting/2005/01/reportdefinition/>

Io ho usato la 2010 e mi sono trovato bene senza riscontrare alcun tipo di problema.

In attesa che la inseriscano una versione funzionante della 2016

<http://schemas.microsoft.com/sqlserver/reporting/2016/01/reportdefinition>

Anche se… <https://msdn.microsoft.com/en-us/library/mt704772(v=sql.105).aspx>

### DataSources

Il Report.DataSources specifica il datasource da usare per il report. È un elemento opzionale.

**<DataSources>**

<DataSource Name="String"> Identificatore del DataSource

<Transaction>Boolean<Transactoin> (opzionale) Specifica se i dataset che usano questo datasource devono essere eseguiti in una unica transazione

<ConnectionProperties> Specifica le informazioni riguardanti come connettersi al DataSource

<DataProvider>String</DataProvider> Specifica le estensioni di data processing usate per il datasource

<ConnectString>String</ConnectString> Specifica le informazioni necessarie per connettersi ad un DataProvider (obbligatorio solo per i serverReport)

<IntegratedSecurity>Boolean</IntegratedSecurity> (su true) Indica che la connessione al datasource deve avvenire usando la sicurezza integrata (inutile nei local report dato che il datasource lo si passa da C#...)

<Prompt>String</Prompt> (opzionale) Specifies the text to use when prompting the user to provide the value or values for a ReportParameter. Prompt element is optional. If this element is not present, the user SHOULD NOT be prompted for, or otherwise be allowed to provide, a value for this parameter.

</ConnectionProperties>

</DataSource>

**</DataSources>**

Codice esempio

  <DataSources>

  <DataSource Name="DataSource1">

  <ConnectionProperties>

  <DataProvider>SQL</DataProvider>

  <ConnectString>data source=dataServer; initial

                       catalog=northwind;</ConnectString>

  <IntegratedSecuity>true</IntegratedSecuity>

  </ConnectionProperties>

  </DataSource>

 </DataSources>

### Datasets

Specifica i dataset usati per il report. È un elemento opzionale.

**<DataSets>**

...

<DataSet Name="String"> Identificatore del Dataset

<[Fields](#Fields)>...</Fields> (Opzionale) Collezione di campi inclusi nel DataSet

<[Query](#Query)>...</Query> (Opzionale) Specifica la query necessaria per prendere le informazioni dal datasource (inutile per i local report)

<CaseSensitivity>Auto,True,False</CaseSensitivity> (Opzionale) Indica se i data nel DataSet sono case-sensitive (Auto per default, autoderivato dal data provider)

<AccentSensitivity>Auto,True,False</AccentSensitivity> (Opzionale) Come CaseSensitivity

<KanatypeSensitivity>Auto,True,False</KanatypeSensitivity> (Opzionale) Come CaseSensitivity

<WidthSensitivity>Auto,True,False</WidthSensitivity> (Opzionale) Come CaseSensitivity

<[Filters](#Filters)>...</Filters> (Opzionale) filtri da applicare ad ogni rida nel DataSet

</DataSet>

...

**</DataSets>**

**<Fields>**

...

<Field Name="String"> Identificatore del campo

<DataField>String</DataField> (opzionale) Specifica il nome del campo che è ritornato dalla query. Se presente questo, <Value> non deve esseci

<Value>Expression</Value> (opzionale) Valore del campo. L’espressione non deve contenere funzioni di aggregazione o report items. Se questo è presente, il DataField non deve essere specificato

</Field>

...

**</Fields>**

Nel caso dei local report ritengo sia più opportuno usare <Value>, mentre nei server report <DataField>

**<Query> (Inutile sui local report)**

  <DataSourceName>String</DataSourceName> (obbligatorio) il nome del datasource sul quale eseguire la query

  <CommandText>String</CommandText> (obbligatorio) Specifica la query da eseguire per ottonere i dati per un DataSet

  <QueryParameters> (opzionale) lista di parametri passati al datasource come parte della query

...

  <QueryParameter Name="@NomeParametro"> (obbligatorio) identificatore del parametro

  <Value>=Expression</Value> (obbligatorio) specifies an expression that evaluates

to the value to pass to a data source when executing

the current Query. The expression may refer to

ReportParameters, but it MUST NOT contain references

to report items or fields, and it MUST NOT use

aggregate functions.

  </QueryParameter>

  ...

  </QueryParameters>

**</Query>**

**<Filters>**

...

<Filter Name="String">

<FilterValues> (Obbligatorio) Valori da usare per il filtro

<FilterValue>Expression</FilterValue> Nota\*

</FilterValues>

<FilterExpression>Expression</FilterExpression> (obbligatorio) Espressione valutata per ogni istanza di un gruppo o per ogni riga associata al Dataset

<Operator>String\*</Operator> (obbligatorio) Operatore di comparazione

</FilterName>

...

**</Filters>**

Nota\*

* If **Filter.Operator** is not set to "In" or "Between", there MUST be exactly one **FilterValues.FilterValue** element.
* If **Filter.Operator** is set to "TopN" or "BottomN", the value of the **FilterValues.FilterValue** element MUST be an Integer or an expression that evaluates to an **Integer**.
* If **Filter.Operator** is set to "TopPercent" or "BottomPercent", the value of the **FilterValues.FilterValue** element MUST be a Float or an expression that evaluates to a **Float**.
* If **Filter.Operator** is set to "Between", there MUST be exactly two **FilterValue** elements.

String\*

* **Equal**: Specifies an equality comparison.
* **Like**: Specifies a like comparison (that is, it compares a string against a pattern).
* **NotEqual**: Specifies an inequality comparison.
* **GreaterThan**: Specifies a comparison to determine whether the **Filter.FilterExpression** value is greater than the **FilterValues** values.
* **GreaterThanOrEqual**: Specifies a comparison to determine whether the **Filter.FilterExpression** value is greater than or equal to the **FilterValues** values.
* **LessThan**: Specifies a comparison to determine whether the **Filter.FilterExpression** value is less than the **FilterValues** values.
* **LessThanOrEqual**: Specifies a comparison to determine whether the **Filter.FilterExpression** value is less than or equal to the **FilterValues** values.
* **TopN**: Specifies whether the **Filter.FilterExpression** value is within the top N data values, where N is specified by FilterValues.FilterValue.
* **BottomN**: Specifies whether the **Filter.FilterExpression** value is within the bottom N data values, where N is specified by **FilterValues.FilterValue**.
* **TopPercent**: Specifies whether the **Filter.FilterExpression** value is within the top N percent of data values, where N is specified by **FilterValues.FilterValue**.
* **BottomPercent**: Specifies whether the **Filter.FilterExpression** value is within the bottom N percent of data values, where N is specified by **FilterValues.FilterValue**.
* **In**: Specifies whether the value of **Filter.FilterExpression** is equal to any **FilterValues.FilterValue** instances.
* **Between**: Specifies whether the value of **Filter.FilterExpression** is between the values of two **FilterValues.FilterValue** instances.

Codice esempio

<DataSets>

  <DataSet Name="DataSet1">

  <Fields>

  <Field Name="ProductID">

  <DataField>ProductID</DataField>

  </Field>

. . .

  </Fields>

  <Query>

  <DataSourceName>DataSource1</DataSourceName>

  <CommandText>SELECT \* From Products WHERE UnitsInStock &gt;= @UnitsInStockMin AND UnitsInStock &lt;= @UnitsInStockMax;</CommandText>

  <QueryParameters>

  <QueryParameter Name="@UnitsInStockMin">

  <Value>=Parameters!UnitsInStockMin.Value</Value>

  </QueryParameter>

  <QueryParameter Name="@UnitsInStockMax">

  <Value>=Parameters!UnitsInStockMax.Value</Value>

  </QueryParameter>

  </QueryParameters>

  </Query>

  </DataSet>

 </DataSets>

### ReportSections

Elemento obbligatorio che deve contenere almeno 1 ReportSection, ossia un elemento che specifica la struttura ed il layout.

**<ReportSections>**

<ReportSection> (obbligatorio) Specifila la struttura e le informazioni di layout

<Body> (obbligatorio) Descrive la struttura e il layout del corpo del ReportSection

<[ReportItems](#ReportItems)> (opzionale) Collezione di ReportItem usati per definire i contenuti di uno spazio di un report

...

</ReportItems>

<Height>RdlSize</Height> (obbligatorio) Indica l’altezza del corpo

<[Style](#Style)> (opzionale) Stile per il corpo del report

...

</Style>

<Page> (obbligatorio) Specifica le informazioni del layout di pagina di un ReportSection

<[PageHeader](#Page_Header_Footer)>...</PageHeader>(opzionale) Header mostrato in alto del ReportSection

<[PageFooter](#Page_Header_Footer)>...</PageFooter> Identico ad Header

  <PageWidth>RdlSize</PageWidth> (8.5in default) Indica la larghezza per il rendering di un ReportSection

<PageHeight>RdlSize</PageHeight> (11in default) Indica l’altezza per il rendering di un ReportSection

<InteractiveWidth>RdlSize</InteractiveWidth> (opzionale) Indica la larghezza per il rendering del Report in un render interattivo

<LeftMargin>RdlSize</LeftMargin> (opzionale) indica la larghezza del margine sinitro

<RightMargin>RdlSize</RightMargin> come LeftMargin, destro

<TopMargin>RdlSize</TopMargin> come LeftMargin, superiore

  <BottomMargin>RdlSize</BottomMargin> come LeftMargin, inferiore

<[Style](#Style)>...</Style>

</Page>

**</ReportSection>**

**<Style>**

  <Border> (opzionale) Indica le proprietà del bordo di un Item. Ignorato se applicato a Paragraph, TextRun.

<Color>String</Color> (opzionale) Indica il colore del bordo

<Style>String\*</Style> (opzionale) Stile del bordo

<Width>RdlSize</Width> (default 1pt) Larghezza del bordo

</Border>

<TopBorder>Border</TopBorder> (opzionale) Childs uguali a border

<BottomBorder>Border</BottomBorder> (opzionale) Childs uguali a border

<RightBorder>Border</RightBorder> (opzionale) Childs uguali a border

<LeftBorder>Border</LeftBorder> (opzionale) Childs uguali a border

<BackgroundColor>String</BackgroundColor> (opzionale) Sfondo dell’Item

<BackgroundGradientType>String</BackgroundGradientType> (opzionale) Gradiante dello sfondo

<BackgroundGradientEndColor>String</BackgroundGradientEndColor> (opzionale) Indica la fine del colore per il gradiante dello sfondo

<BackgroundHatchType>String\*\*</BackgroundHatchType> (opzionale) Element specifies the hatching pattern of a report item

<[BackgroundImage](#BackgroundImage)>...</BackgroundImage> (opzionale) Immagine di sfondo

<FontStyle>Default,normal,italic</FontStyle> (opzionale) Stile del fondo

<FontFamily>String</FontFamily> (opzionale) Nome del FontFamily del testo del report item

<FontSize>RdlSize</FontSize> (opzionale, <200pt) Grandezza del font del testo del report item

<FontWeight>String\*\*\*</FontWeight> (opzionale) specifies the thickness of font for text within a report item

<Format>String</Format> (opzionale) Formattazione del valore mostrato nel testo

<TextDecoration>String\*\*\*\*</TextDecoration> (opzionale) element specifies any special

text formatting for text in a report item

<TextAlign>String</TextAlign> (opzionale) **Default**: Specifies the default horizontal text alignment, which is the same as "General". **General**: Specifies that the text alignment is derived from the data type and the peer Style.Direction element. **Left**: Specifies that the text is left-aligned. **Center**: Specifies that the text is center-aligned. **Right**: Specifies that the text is right-aligned.

<TextEffect>String</TextEffect> (opzionale) **Default**: Specifies the default text effect, which is the same as "None". **None**: Specifies that no special effect is applied to the text. **Shadow**: Specifies that the text has a shadow. **Emboss**: Specifies that the text is embossed. **Embed**: Specifies that the text has an embedded effect. **Frame**: Specifies that a frame is added around the text.

<VerticalAlign>String</VerticalAlign> (opzionale) **Default**: Specifies the default vertical text alignment, which is the same as "Top". **Top**: Specifies top-aligned text. **Middle**: Specifies vertically-centered text alignment. **Bottom**: Specifies bottom-aligned text.

<Color>String</Color> (opzionale)

<ShadowColor>String</ShadowColor> (opzionale)

<ShadowOffset>String</ShadowOffset> (opzionale)

<PaddingLeft>RdlSize</PaddingLeft> (opzionale, <1000pt) Padding tra il bordo sinistro dell’Item ed il suo contenuto

<PaddingRight>RdlSize</PaddingRight> Come PaddingLeft

<PaddingTop>RdlSize</PaddingTop> Come PaddingLeft

<PaddingBottom>RdlSize</PaddingBottom> Come PaddingLeft

**</Style>**

String\*

**None**: Specifies that no border is drawn. **Dotted**: Specifies a dotted line for a border. **Dashed**: Specifies a dashed line for a border. **Solid**: Specifies a solid line for a border. **Double**: Specifies a double solid line for a border. **DashDot**: Specifies a dash-dot pattern line for a border. This value MUST be permitted only within a [Chart](https://msdn.microsoft.com/en-us/library/ee209880(v=sql.105).aspx) element. **DashDotDot**: Specifies a dash-dot-dot pattern line for a border. This value MUST be permitted only within a **Chart** element. If the **Border.Style** element is not present, its value is interpreted as "Default".

String\*\*

**Default**: Specifies the default background hatching pattern, which is treated the same as "None". **None**: Specifies that there is no background hatching. **BackwardDiagonal**: Specifies a backward-diagonal hatching pattern. **Cross**: Specifies a cross-hatching pattern. **DarkDownwardDiagonal**: Specifies a dark downward-diagonal hatching pattern. **DarkHorizontal**: Specifies a dark horizontal hatching pattern. **DarkUpwardDiagonal**: Specifies a dark upward-diagonal hatching pattern. **DarkVertical**: Specifies a dark vertical hatching pattern.

**DashedDownwardDiagonal**: Specifies a dashed downward-diagonal hatching pattern. **DashedHorizontal**: Specifies a dashed horizontal hatching pattern. **DashedUpwardDiagonal**: Specifies a dashed upward-diagonal hatching pattern. **DashedVertical**: Specifies a dashed vertical hatching pattern. **DiagonalBrick**: Specifies a diagonal brick hatching pattern. **DiagonalCross**: Specifies a diagonal cross-hatching pattern. **Divot**: Specifies a divot hatching pattern. **DottedDiamond**: Specifies a dotted diamond hatching pattern. **DottedGrid**: Specifies a dotted grid hatching pattern. **ForwardDiagonal**: Specifies a forward-diagonal hatching pattern. **Horizontal**: Specifies a horizontal hatching pattern. **HorizontalBrick**: Specifies a horizontal brick hatching pattern. **LargeCheckerBoard**: Specifies a large checkerboard hatching pattern. **LargeConfetti**: Specifies a large confetti hatching pattern. **LargeGrid**: Specifies a large grid hatching pattern. **LightDownwardDiagonal**: Specifies a light downward-diagonal hatching pattern. **LightHorizontal**: Specifies a light horizontal hatching pattern. **LightUpwardDiagonal**: Specifies a light upward-diagonal hatching pattern. **LightVertical**: Specifies a light vertical hatching pattern. **NarrowHorizontal**: Specifies a narrow horizontal hatching pattern. **NarrowVertical**: Specifies a narrow vertical hatching pattern. **OutlinedDiamond**: Specifies an outlined diamond hatching pattern. **Percent05**: Specifies a 5-percent hatching pattern. **Percent10**. **Percent20**. **Percent25**. **Percent30**. **Percent40**. **Percent50**. **Percent60**. **Percent70**. **Percent75**. **Percent80**. **Percent90**. **Plaid**: Specifies a plaid hatching pattern. **Shingle**: Specifies a shingle hatching pattern. **SmallCheckerBoard**: Specifies a small checkerboard hatching pattern. **SmallConfetti**: Specifies a small confetti hatching pattern. **SmallGrid**: Specifies a small grid hatching pattern. **SolidDiamond**: Specifies a solid diamond hatching pattern. **Sphere**: Specifies a sphere hatching pattern. **Trellis**: Specifies a trellis hatching pattern. **Vertical**: Specifies a vertical hatching pattern. **Wave**: Specifies a wave hatching pattern. **Weave**: Specifies a weave hatching pattern. **WideDownwardDiagonal**: Specifies a wide downward-diagonal hatching pattern. **WideUpwardDiagonal**: Specifies a wide upward-diagonal hatching pattern. **ZigZag**: Specifies a zigzag hatching pattern.

String\*\*\*

**Default**: Specifies the default font thickness, which is the same as "Normal". **Thin**: Specifies a font thickness that matches cascading style sheets (CSS) font weight 100. **ExtraLight**: Specifies a font thickness that matches CSS font weight 200. **Light**: Specifies a font thickness that matches CSS font weight 300. **Normal**: Specifies a font thickness that matches CSS font weight 400. **Medium**: Specifies a font thickness that matches CSS font weight 500. **SemiBold**: Specifies a font thickness that matches CSS font weight 600. **Bold**: Specifies a font thickness that matches CSS font weight 700. **ExtraBold**: Specifies a font thickness that matches CSS font weight 800. **Heavy**: Specifies a font thickness that matches CSS font weight 900.

String\*\*\*\*

**Default**: Specifies the default text formatting, which is the same as "None".

**None**: Specifies that there is no text formatting.

**Underline**: Specifies that the text is underlined.

**Overline**: Specifies that the text has an overline.

**LineThrough**: Specifies that the text has a strikethrough.

**<BackgroundImage Name=**”String”**>**

<Source>Stringa</Source> (obbligatorio) **External**: Specifies that the peer BackgroundImage.Value element MUST contain a String constant or an expression that evaluates to the location of an image. **Embedded**: Specifies that the peer BackgroundImage.Value element MUST contain a String constant or an expression that evaluates to the name of an EmbeddedImage within a report. **Database**: Specifies that the peer BackgroundImage.Value element MUST contain an expression (such as a field in the database) that evaluates to the binary data for an image.

<Value>String\*</Value> (obbligatorio) Path immagine

<MIMEType>String</MIMEType> (opzionale) Formato dell’immagine

<BackgroundRepeat>String</BackgroundRepeat> (opzionale) **Default**: Specifies the default BackgroundImage repeating behavior. **Repeat**: Specifies that the image should repeat both horizontally and vertically to fill the space. **RepeatX**: Specifies that the image should repeat horizontally to fill the space. This value MUST NOT be allowed if the containing report item is a chart. **RepeatY**: Specifies that the image should repeat vertically to fill the space. This value MUST NOT be allowed if the containing report item is a chart. **Fit**: Specifies that the image should stretch to fill the space. This value MUST be allowed only when the containing report item is a chart. Clip: Specifies that the image should be clipped to the available space. **FitProportional**: Specifies that the image should stretch to fill the space and, at the same time, preserve its aspect ratio.

<Position>String</Position> (opzionale) **Default**: Specifies the default position of the background images, which is the same as "Center" for charts. **Top**: Specifies that the background image should be drawn at the top center of the containing report item. **TopLeft**: Specifies that the background image should be drawn at the top left of the containing report item. **TopRight**: Specifies that the background image should be drawn at the top right of the containing report item. **Left**: Specifies that the background image should be drawn at center left of the containing report item. **Center**: Specifies that the background image should be drawn at the center of the containing report item. **Right**: Specifies that the background image should be drawn at the center right of the containing report item. **BottomRight**: Specifies that the background image should be drawn at the bottom right of the containing report item. **Bottom**: Specifies that the background image should be drawn at the bottom center of the containing report item. **BottomLeft**: Specifies that the background image should be drawn at the bottom left of the containing report item.

**</BackgroundImage>**

String\*

If the peer BackgroundImage.Source element is set to "External", the value of the BackgroundImage.Value element MUST be a String or an expression that evaluates to the location of an image. This location MUST be a ReportPath or RdlURL value.

If the peer BackgroundImage.Source element is set to "Embedded", the value of the BackgroundImage.Value element MUST be a String or an expression that evaluates to the name of an EmbeddedImage in a report.

If the peer BackgroundImage.Source element is set to "Database", the value of the BackgroundImage.Value element MUST be a String or an expression that evaluates to the binary data for an image.

If the BackgroundImage.Value element has an empty value, an image MUST NOT be displayed.

**<PageHeader>** (opzionale) Header mostrato in alto del ReportSection

<Height>RdlSize</Height> (obbligatorio) Altezza dell’Header

<PrintOnFirstPage>Boolean</PrintOnFirstPage> (opzionale) Stampa sulla prima pagina

<PrintOnLastPage>Boolean</PrintOnLastPage> (opzionale) Stampa sull’ultima pagina

<PrintBetweenSections>Boolean</PrintBetweenSections> (opzionale) Specifies whether the PageSection should be shown on or a ReportSection

<[ReportItems](#ReportItems)>...</ReportItems>

<[Style](#Style)>...</Style>

**</PageHeader>**

Codice esempio

<ReportSections>

  <ReportSection>

  <Body>

  <ReportItems>...</ReportItems>

  <Height>2.5in</Height>

  <Style>

  <BackgroundColor>LightBlue</BackgroundColor>

  </Style>

  </Body>

  <Width>6.5in</Width>

  <Page>

  <PageHeader>

  <Height>0.5in</Height>

  <PrintOnFirstPage>true</PrintOnFirstPage>

  <PrintOnLastPage>true</PrintOnLastPage>

  <PrintBetweenSections>true</PrintBetweenSections>

  <ReportItems>...</ReportItems>

  <Style>

  <BackgroundColor>Silver</BackgroundColor>

  </Style>

  </PageHeader>

  <PageFooter>

  <Height>0.75in</Height>

  <PrintOnFirstPage>true</PrintOnFirstPage>

  <PrintOnLastPage>true</PrintOnLastPage>

  <PrintBetweenSections>true</PrintBetweenSections>

  <ReportItems>...</ReportItems>

  <Style>

  <BackgroundColor>LightGrey</BackgroundColor>

  </Style>

  </PageFooter>

  <PageWidth>8.5in</PageWidth>

  <PageHeight>5in</PageHeight>

  <InteractiveWidth>8.5in</InteractiveWidth>

  <LeftMargin>0.5in</LeftMargin>

  <RightMargin>0.5in</RightMargin>

  <TopMargin>0.25in</TopMargin>

  <BottomMargin>0.25in</BottomMargin>

  <Style>

  <Border>

  <Color>Black</Color>

  <Style>Solid</Style>

  <Width>1pt</Width>

  </Border>

  </Style>

  </Page>

  </ReportSection>

  <ReportSection>

  <Body>

  <ReportItems>...</ReportItems>

  <Height>3in</Height>

  <Style>

  <BackgroundColor>Tan</BackgroundColor>

  </Style>

  </Body>

  <Width>6.5in</Width>

  <Page>

  <PageHeader>

  <Height>0.5in</Height>

  <PrintOnFirstPage>true</PrintOnFirstPage>

  <PrintOnLastPage>true</PrintOnLastPage>

  <PrintBetweenSections>true</PrintBetweenSections>

  <ReportItems>

  </ReportItems>

  <Style>

  <BackgroundColor>LightSteelBlue</BackgroundColor>

  </Style>

  </PageHeader>

  <PageFooter>

  <Height>0.75in</Height>

  <PrintOnFirstPage>true</PrintOnFirstPage>

  <PrintOnLastPage>true</PrintOnLastPage>

  <PrintBetweenSections>true</PrintBetweenSections>

  <ReportItems>...</ReportItems>

  <Style>

  <BackgroundColor>LightBlue</BackgroundColor>

  </Style>

  </PageFooter>

  <PageWidth>8.5in</PageWidth>

  <PageHeight>5in</PageHeight>

  <InteractiveWidth>8.5in</InteractiveWidth>

  <LeftMargin>0.5in</LeftMargin>

  <RightMargin>0.5in</RightMargin>

  <TopMargin>0.25in</TopMargin>

  <BottomMargin>0.25in</BottomMargin>

  <Style>

  <Border>

  <Color>Black</Color>

  <Style>Solid</Style>

  <Width>1pt</Width>

  </Border>

  </Style>

  </Page>

  </ReportSection>

</ReportSections>

Codice esempio

<Style>

<Border>

<Color>Black</Color>

<Style>Solid</Style>

<Width>1pt</Width>

</Border>

<BackgroundColor>LightSteelBlue</BackgroundColor>

</Style>

Codice esempio

<PageHeader>

<Height>0.5in</Height>

<PrintOnFirstPage>true</PrintOnFirstPage>

<PrintOnLastPage>true</PrintOnLastPage>

<PrintBetweenSections>true</PrintBetweenSections>

<ReportItems>...</ReportItems>

<Style>

<BackgroundColor>LightSteelBlue</BackgroundColor>

</Style>

</PageHeader>

### ReportParameters

Parametri da passare al report in input (solo server report, per i local non ha molto senso in quanto ci si aspetta che i datasets passati siano già filtrati…)

**<ReportParameters>** (opzionale, solo 1 volta)

  <ReportParameter Name="String"> (obbligatorio)

  <DataType>VediNota\*</DataType> (obbligatorio, solo 1 volta) Specifica il tipo di dato

  <Nullable>Boolean</Nullable> (opzionale, default=false, solo 1 volta) Specifica se il valore può essere null. Se MultiValue = true, Nullable deve essere = false

  <DefaultValue> (opzionale, solo 1 volta) Specifica valori di default non forniti dall’utente. Obbligatorio se Prompt è non specificato AND Nullable=false or ValidValues non conviene null

  <DataSetReference> (opzionale, obbligatorio se Values non è specificato, solo 1 volta) Dataset di riferimento per otterenere i valori di default

  <DataSetName>String</DataSetName> (obbligatorio, solo 1 volta) Nome del Dataset

  <ValueField>String</ValueField> (obbligatorio, solo 1 volta) Valore del campo

  <LabelField>String</LabelField> (opzionale, solo 1 volta) specifies the name of the field in the referenced DataSet from which values are retrieved to populate the labels of a parameter’s DefaultValue

  </DataSetReference>

<Values> (opzionale, solo 1 volta) collezione di valori di default

<Value>String</Value> (obbligatorio) valori di default

</Values>

  </DefaultValue>

  <AllowBlank>Boolean</AllowBlank> (opzionale, default=false, solo 1 volta) ignorato se DataType != String. Consente stringhe vuote

  <Prompt>String</Prompt> (opzionale, solo 1 volta) element specifies the text to use when prompting the user to provide the value or values for a ReportParameter.

  <ValidValues> (opzionale, solo 1 volta) valori accettati dal report

  <DataSetReference> (opzionale, obbligatorio se ParameterValues non è stato specificato, solo 1 volta) Indica il dataset di riferimento per i valori validi

  <DataSetName>String</DataSetName> (obbligatorio, solo 1 volta) Nome del Dataset

  <ValueField>String</ValueField> (obbligatorio, solo 1 volta) Valore del campo

  <LabelField>String</LabelField> (opzionale, solo 1 volta) specifies the name of the field in the referenced DataSet from which values are retrieved to populate the labels of a parameter’s ValidValues

  </DataSetReference>

<ParameterValues> (opzionale, solo 1 volta) collezione di valori accettati

<ParameterValue> (obbligatorio) valore accettato

<Value>String</Value> (opzionale, solo 1 volta) Specifica il valore del parametro accettato

<Label>String</Label> (opzionale, solo 1 volta) Specifica l’etichetta per descrivere il valore accettato

</ParameterValue>

</ParameterValues>

  </ValidValues>

  <Hidden>Boolean</Hidden> (opzionale, default=false, solo 1 volta) Specifica se il parametro deve essere nascosto

  <MultiValue>Boolean</MultiValue> (opzionale, default=false, solo 1 volta) Specifica se il reportparameter accetta più valori. Ignorato se DataType = Boolean

  <UsedInQuery>String</UsedInQuery> (opzionale, solo 1 volta) **Auto** (default): The value of the ReportParameter.UsedInQuery element is interpreted as true if any QueryParameter value expression is a simple reference to this ReportParameter, if there is a Subreport in the report, or if there exists any QueryParameter value expression that is anything but a constant or a simple ReportParameter reference. Otherwise, the value of the ReportParameter.UsedInQuery element is interpreted as false. **True**: The ReportParameter is used in a query in the report. **False**: The ReportParameter is not used in any query in the report.

  </ReportParameter>

**</ReportParameters>**

VediNota\*

**Boolean**: The value or values for [ReportParameter.ValidValues](https://msdn.microsoft.com/en-us/library/ee209138(v=sql.105).aspx) and [ReportParameter.DefaultValue](https://msdn.microsoft.com/en-us/library/ee209507(v=sql.105).aspx) and the value or values specified by the user for the **ReportParameter** MUST be "Boolean".

**DateTime**: The value or values for **ReportParameter.ValidValues** and **ReportParameter.DefaultValue** and the value or values that are specified by the user for the **ReportParameter** MUST all be "DateTime".

**Integer**: The value or values for **ReportParameter.ValidValues** and **ReportParameter.DefaultValue** and the value or values specified by the user for the **ReportParameter** MUST all be "Integer".

**Float**: The value or values for **ReportParameter.ValidValues** and **ReportParameter.DefaultValue** and the value or values specified by the user for the **ReportParameter** MUST all be "Float".

**String**: The value or values for **ReportParameter.ValidValues** and **ReportParameter.DefaultValue** and the value or values specified by the user for the **ReportParameter** MUST all be "String".

Codice esempio

<ReportParameters>

  <ReportParameter Name="Product">

  <DataType>String</DataType>

  <DefaultValue>

  <DataSetReference>

  <DataSetName>DataSet1</DataSetName>

  <ValueField>ProductID</ValueField>

  </DataSetReference>

  </DefaultValue>

  <Prompt>Product</Prompt>

  <ValidValues>

  <DataSetReference>

  <DataSetName>DataSet1</DataSetName>

  <ValueField>ProductID</ValueField>

  <LabelField>ProductName</LabelField>

  </DataSetReference>

  </ValidValues>

  <MultiValue>true</MultiValue>

  </ReportParameter>

</ReportParameters>

### ReportItems

“The **ReportItems** element specifies the collection of [report items](https://msdn.microsoft.com/en-us/library/ee240603(v=sql.105).aspx#gt_c6f8e999-fca9-4e79-96e7-fb4c2c43d601) used to define the contents of a region of a [Report](https://msdn.microsoft.com/en-us/library/ee224052(v=sql.105).aspx).”

**<ReportItems>**

<Tablix Name="String">...</Tablix>

<Textbox Name="String">...</Textbox>

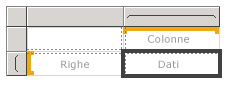
<Rectangle Name="String">...</Rectangle>

<Subreport Name="String">...</Subreport>

<Image Name="String">...</Image>

**</ReportItems>**

### Tablix



Dalla definizione MS (<https://msdn.microsoft.com/en-us/library/ee224655(v=sql.105).aspx)>

The **Tablix** element defines a [tablix](https://msdn.microsoft.com/en-us/library/ee240603(v=sql.105).aspx#gt_f9f5d4be-2a9e-4556-90f6-d4ed1678f0b4), which is a layout grid that consists of columns and rows that can optionally be filtered, sorted, grouped, nested, and repeated.

An element of type **Tablix** MUST NOT be specified within a page header or a page footer.

There are two collections of [TablixMember](https://msdn.microsoft.com/en-us/library/ee223786(v=sql.105).aspx) elements. **TablixMember** elements that have a [TablixRowHierarchy](https://msdn.microsoft.com/en-us/library/ee208756(v=sql.105).aspx) ancestor in the tablix are called tablix [row members](https://msdn.microsoft.com/en-us/library/ee240603(v=sql.105).aspx#gt_85da4bce-7983-4ac7-98ca-9cf698d74881). **TablixMember** elements that have a [TablixColumnHierarchy](https://msdn.microsoft.com/en-us/library/ee240038(v=sql.105).aspx) ancestor in the tablix are called tablix [column members](https://msdn.microsoft.com/en-us/library/ee240603(v=sql.105).aspx#gt_0e316a29-1401-442d-96ce-bdf521b18564).

The quantity of [TablixColumn](https://msdn.microsoft.com/en-us/library/ee240523(v=sql.105).aspx) elements within the tablix MUST equal the quantity of tablix column members that do not have a **TablixMember** element descendant. The quantity of **TablixColumn** elements within the tablix MUST equal the quantity of [TablixCell](https://msdn.microsoft.com/en-us/library/ee209062(v=sql.105).aspx) elements within the tablix.

The value of the [Tablix.LayoutDirection](https://msdn.microsoft.com/en-us/library/ee209751(v=sql.105).aspx) element changes the effect that the value of the [Tablix.GroupsBeforeRowHeaders](https://msdn.microsoft.com/en-us/library/ee209061(v=sql.105).aspx) element will have. This is described under **Tablix.GroupsBeforeRowHeaders**.

Columns of tablix [row headers](https://msdn.microsoft.com/en-us/library/ee240603(v=sql.105).aspx#gt_4a2f606e-7699-46fb-bc95-82a9e6dae94f) MUST be laid out before columns that are defined by **TablixColumn** elements, and they MUST be considered in conjunction with the **Tablix.GroupsBeforeRowHeaders** element.

The [Tablix.TablixCorner](https://msdn.microsoft.com/en-us/library/ee224144(v=sql.105).aspx) element MUST be specified if the following conditions are true:

* The tablix has a **TablixRowHierarchy** element child that has a [TablixHeader](https://msdn.microsoft.com/en-us/library/ee240507(v=sql.105).aspx) element descendant.
* The tablix has a **TablixColumnHierarchy** element child that has a **TablixHeader** element descendant.

If either the **TablixRowHierarchy** element or the **TablixColumnHierarchy** element does not have a **TablixHeader** element descendant, the **Tablix** element MUST NOT have a child [TablixCorner](https://msdn.microsoft.com/en-us/library/ee209608(v=sql.105).aspx) element.

If a [TablixCornerRow](https://msdn.microsoft.com/en-us/library/ee208749(v=sql.105).aspx) element is specified, the quantity of descendant [TablixCornerCell](https://msdn.microsoft.com/en-us/library/ee224070(v=sql.105).aspx) elements in the tablix MUST equal the quantity of unique cumulative widths for each tablix row member of the tablix. "Cumulative width" for a tablix row member is defined as the sum of the values of all [TablixHeader.Size](https://msdn.microsoft.com/en-us/library/ee240085(v=sql.105).aspx) elements that are descendant of the tablix row member.

If a **TablixCorner** element is specified, the quantity of descendant **TablixCornerRow** elements in the tablix MUST equal the quantity of unique cumulative heights for each tablix column member of the tablix. "Cumulative height" for a tablix column member is defined as the sum of the values of all **TablixHeader.Size** elements descendant of the tablix column member.

Borders and any [background images](https://msdn.microsoft.com/en-us/library/ee240603(v=sql.105).aspx#gt_bc67ce68-380f-4fe4-987e-6d3b0d795635) that are specified under [Tablix.Style](https://msdn.microsoft.com/en-us/library/ee224364(v=sql.105).aspx) can be repeated on vertical and horizontal page breaks by using the [Tablix.OmitBorderOnPageBreak](https://msdn.microsoft.com/en-us/library/ee209494(v=sql.105).aspx) element.

  <Tablix Name="String"> (obbligatorio) Identificatore

<TablixCorner> Definisce i contenuti dell’intersezione degli headers dei gruppi di righe e dei gruppi di colonne.

  <TablixCornerRows> (obbligatorio, solo 1 volta) Specifica una collezione di righe

  <TablixCornerRow> Specifica una riga in un TablixCorner. Se è specificato un TablixCorner, la quantità di TablixConerRow in una Tablix deve essere uguale alla quantità di altezze cumulative per ogni tablix column member della Tablix. L’altezza cumulativa per un tablix column member è definita come la somma di tutti i valori TablixHeader.Size discendenti del tablix column member.

  <TablixCornerCell> (opzionale) Specifica le celle della riga di un TablixCorner. Se è stata definita una TablixCornerRow, la quantità di TablixCornerCell nella Tablix deve essere uguale alla quantità di larghezze cumulative per ogni tablix row member della Tablix. La larghezza cumulativa per un tablix row member è definita come la somma di tutti i valori TablixHeader.Size discententi del tablix row member.

  <[CellContents](#CellContents)>...<CellContents> Definisce il contenuto della cella del TablixCornerCell. The TablixCornerCell.CellContents element MUST NOT be specified if the position in the corner is covered by a span from another cell. Otherwise, this element MUST be specified exactly once.

The value of the CellContents.ColSpan child element, decremented by 1, specifies the quantity of sequential sibling TablixCornerCell elements that MUST NOT have a child TablixCornerCell.CellContents element specified.

The value of the CellContents.RowSpan child element, decremented by 1, specifies the quantity of sequential TablixCornerRow element siblings to the ancestor TablixCornerRow element that MUST NOT have a child TablixCornerCell.CellContents element specified in the same positional order.

  </TablixCornerCell>

  </TablixCornerRow>

  </TablixCornerRows>

</TablixCorner>

  <TablixColumnHierarchy> (obbligatorio, solo 1 volta) Specifica la gerarchia sulla colonna di una Tablix

  <TablixMembers> (obbligatorio, solo 1 volta) Specifica i membri della TablixHierarchy

  <[TablixMember](#TablixMembers)>...</TablixMember> (opzionale) L'elemento padre TablixMember definisce la struttura di una gerarchia Tablix.

  </TablixMembers>

  </TablixColumnHierarchy>

  <TablixRowHierarchy> (obbligatorio, solo 1 volta) Specifica la gerarhia sulla riga di una Tablix

  <[TablixMembers](#TablixMembers)>...</TablixMembers>

  </TablixRowHierarchy>

  <TablixBody> Specifica il corpo di una Tablix. Il TablixBody deve essere specificato se la tablix ha sia un TablixRowHierarchy con un TablixHeader e una TablixColumnHierarchy con un TablixHeader.

  <TablixColumns> (obbligatorio, solo 1 volta) Definisce la collezione di colonne del corpo di una Tablix

  <TablixColumn> Definisce una colonna nel corpo di una Tablix. The quantity of TablixColumn elements MUST equal the quantity of tablix column members that do not have a TablixMember element descendant

  <Width>RdlSize</Width> (obbligatorio, solo 1 volta) Specifica la larghezza della colonna in una Tablix

  </TablixColumn>

  </TablixColumns>

  <TablixRows> (obbligatorio, solo 1 volta) Definisce la collezione di righe del corpo di una Tablix

  <TablixRow> Definisce una riga nel corpo di una Tablix. The quantity of TablixRow elements MUST equal the quantity of TablixMember elements in the TablixRowHierarchy that do not have a TablixMember element descendant.

  <Height>RdlSize</Height> (obbligatorio, solo 1 volta) Specifica l’altezza della riga in una Tablix

  <TablixCells> (obbligatorio, solo 1 volta) Specifica una collezione di TablixCell in una riga di una Tablix

  <TablixCell> Definisce il contenuto di una cella nel corpo di una Tablix

  <[CellContents](#CellContents)>... </CellContents> Definisce il contenuto della cella di una Tablix, e può contenere 0 or 1 ReportItem

  </TablixCell>

  </TablixCells>

  </TablixRow>

  </TablixRows>

  </TablixBody>

<RepeatColumnHeaders>Boolean</RepeatColumnHeaders> (opzionale, 1 sola volta, default=false) Ripete gli Headers per ogni pagina attraversata dalla colonna

<RepeatRowHeaders>Boolean</RepeatRowHeaders> (opzionale, 1 sola volta, default=false) Come RepeatColumnHeaders

<FixedColumnHeaders>Boolean</FixedColumnHeaders> (opzionale, 1 sola volta, default=false) Gli headers rimangono visibili durante lo scroll della pagina

<FixedRowHeaders>Boolean</FixedRowHeaders> (opzionale, 1 sola volta, default=false)

<[Style](#Style)>...</Style>

<Top>RdlSize</Top> (opzionale, 1 sola volta) specifies the distance of the Tablix from the top of the containing object.

<Left>RdlSize</Left> (opzionale, 1 sola volta) come top, sinistra

<Height>RdlSize</Height> (opzionale, 1 sola volta, default=autoderivata) Altezza della Tablix

<Width>RdlSize</Width> (opzionale, 1 sola volta, default=autoderivata) Larghezza della Tablix

  <ZIndex>Integer</ZIndex> (opzionale, 1 sola volta) Lo ZIntex della matrice. “Specifies the drawing order of an item within the containing item. Items are drawn in order according to the zIndex element for the item. Items with lower indices are drawn first, appearing behind items with higher indices.”

<[Visibility](#Visibility)>...</Visibility>

  <ToolTip>String</ToolTip> (opzionalem 1 sola volta) Specifica il tooltip della tablix

<DocumentMapLabel>String</DocumentMapLabel> (opzionale, 1 sola volta, identificatore) Specifica un document map label per identificare la Tablix nel rendered Report

<Bookmark>String</Bookmark> (opzionale, 1 sola volta) Specifica il segnalibro di una Tablix

<[CustomProperties](#CustomProperties)>...</CustomProperties> (Opzionale) specifies properties about which report design tools may pass information to custom report renderers and custom report items.

  <[PageBreak](#PageBreak)>...</PageBreak> (opzionale, 1 sola volta) Interruzione di pagina

  <PageName>String</PageName> (opzionale, 1 sola volta) Nome della pagina impaginata

<KeepTogether>Boolean</KeepTogether> (opzionale, 1 sola volta, default=false) Il render prova a far entrare l’intera Tablix in una pagina

<NoRowsMessage>String</NoRowsMessage> (opzionale, 1 sola volta) Il testo da mostrare quando non ci sono dati disponibili

<DataSetName>String</DataSetName> (opzionale, 1 sola volta) Specifica il nome del DataSet usato per la Tablix. Il valore di questo elemento deve essere un DataSet.Name.

  <[Filters](#Filters)>...</Filters> (opzionale, 1 sola volta) Interruzione di pagina

<DataElementName>String</DataElementName> (opzionale, 1 sola volta) specifica il nome da usare per il data element di una Tablix nel data rendering

  <OmitBorderOnPageBreak>Boolean</OmitBorderOnPageBreak> (opzionale, 1 sola volta, default = false) Specifica se i bordi e le immagini di sfondo specificate in Tablix.Style nell’output devono essere mostrate su pagine multiple

  <DataElementOutput>String\*</DataElementOutput> (opzionale, 1 sola volta) Specifica se la tablix è inclusa nel data rendering.

  </Tablix>

String\*

**Output**: The tablix data appears in the data rendering output.

**NoOutput**: The tablix data does not appear in the data rendering output.

**Auto** (default): If the value of the grandchild Visibility.Hidden element of the **Tablix** element is the **String** literal value of "true" (not an expression) and if a grandchild Visibility.ToggleItem element is not specified for the **Tablix**, the value of **Tablix.DataElementOutput** value is interpreted as "NoOutput".

**<CellContents>** Definisce il contenuto della cella di una Tablix, e può contenere 0 or 1

ReportItem

<Eventuale ReportItem> Può essere un qualsiasi ReportItem. Se volessi mettere N ReportItem, potrei inserire nella cella una “*Tablix*” contentente N celle ognuna delle quali ha un ReportItem… altrimenti aiutarmi con il report Item “*Rettangolo*” …

<DataElementName>String</DataElementName> (opzionale, 1 sola volta, identificatore) The TablixCell.DataElementName element specifies the name to use for the data element of the TablixCell in the output of a data rendering.

<DataElementOutput>String\*</DataElementOutput> (opzionale, 1 sola volta) Specifica l’inclusione della TablixCell in un data rendering

**</CellContents>**

String\*

**ContentsOnly** (default): The **TablixCell** does not appear in the data rendering output, but its contents appear in the data rendering output as if they were in the container of the **TablixCell** element.

**Output**: The **TablixCell** appears in the data rendering output.

**NoOutput**: The **TablixCell** does not appear in the data rendering output, and its contents do not appear in the data rendering output.

**<TablixMembers>** (opzionale) L'elemento padre TablixMember definisce la struttura di una gerarchia Tablix.

  <TablixMember>

  <[Group](#Group)>...</Group> (opzionale, 1 sola volta) The Group element specifies the group for a TablixMember element. If the Group element is specified, it is of type Group. If a TablixMember element has a detail member ancestor, this element MUST NOT be specified.

  <SortExpressions> (opzionale)

  <SortExpression> (Obbligatorio) Espressione di ordinamento

  <Value>Expression</Value>

  </SortExpression>

  </SortExpressions>

<DataElementName>String</DataElementName> (opzionale, 1 sola volta) Specifica il nome da usare per il data element di un tablixmember nel data rendering

<DataElementOutput>String</DataElementOutput> (opzionale, 1 sola volta) Specifica quando un tablix è incluso nel data rendering. Value MUST be one of the following: **Auto** (default): If the parent element is a dynamic member, this value is interpreted as "Output". **Output**: The TablixMember appears in the data rendering output. **NoOutput**: The TablixMember does not appear in the data rendering output.

<FixedData>Boolean</FixedData> (opzionale, 1 sola volta) specifies whether a TablixMember element and its associated TablixBody element cells remain visible when the Tablix is partially scrolled off a page.

<HideIfNoRows>String</HideIfNoRows> (opzionale, 1 sola volta) specifies whether a row or column that is associated with a static member SHOULD be hidden if a tablix contains no rows of data

<KeepTogether>Boolean</KeepTogether> (opzionale, 1 sola volta, default=false) Il render prova a far entrare l’intero member in una pagina

<KeepWithGroup>String</KeepWithGroup> (opzionale, 1 sola volta) element specifies whether an attempt SHOULD be made to keep each rendered instance of a parent static member with the closest non-hidden instance of a preceding or subsequent sibling dynamic member. The value MUST be one of the following: None (default): The TablixMember.KeepWithGroup element is ignored. Before: Rendered instances of the parent static member SHOULD be kept with the closest preceding non-hidden sibling dynamic member. After: Rendered instances of the parent static member SHOULD be kept with the closest subsequent non-hidden sibling dynamic member.

<RepeatOnNewPage>Boolean</RepeatOnNewPage> (opzionale, 1 sola volta)

<TablixHeader>String</TablixHeader> (opzionale, 1 sola volta)

<[TablixMembers](#TablixMembers)>...</TablixMembers> (opzionale, 1 sola volta)

<[Visibility](#Visibility)>...</Visibility> (opzionale, 1 sola volta) Opzioni di visiblitià

<[CustomProperties](#CustomProperties)>...</CustomProperties> (opzionale, 1 sola volta)Proprietà custom

<[TablixHeader](#TablixHeader)>...</TablixHeader> (opzionale, 1 sola volta) Header della cella

  </TablixMember>

**</TablixMembers>**

**<Group Name="String">** (opzionale, 1 sola volta) Specifica le espressioni con le quali vengono raggruppati i dati

  <GroupExpressions> (opzionale, 1 sola volta) Collezione di group expression

  <GroupExpression>Expression</GroupExpression> Espressione di raggruppamento

  </GroupExpressions>

<DataElementName>String</DataElementName> (opzionale, 1 sola volta) Specifica il nome da usare per il data element di un gruppo nel data rendering

<DataElementOutput>String</DataElementOutput> (opzionale, 1 sola volta) Specifica quando un gruppo è incluso nel data rendering. Value MUST be one of the following: Output (default): The Group data appears in the data rendering output. NoOutput: The Group data does not appear in the data rendering output.

<DocumentMapLabel>String</DocumentMapLabel> (opzionale, 1 sola volta) specifies a document map label to identify a Group within the rendered report

<DomainScope>String</DomainScope> (opzionale, 1 sola volta) specifies the name of the scope (data region or group) in which to evaluate the group expressions

<[Filters](#Filters)>...</Filters> (opzionale) collezione di filtri

<PageName>String</PageName> (opzionale) specifies value to use for the name of a paginated page

**</Group>**

**<Visibility>** (opzionale, 1 sola volta) Opzioni di visiblitià

<Hidden>Boolean</Hidden> (opzionale, 1 sola volta, default=false) Specifica

se l’item deve essere inizialmente nascosto

<ToggleItem>String</ToggleItem> (opzionale, 1 sola volta) specifies the name of a

Textbox that is used to hide or unhide the

containing report item. The value of this element

MUST contain the name of a text box in the same

group as the containing report item or in any

containing ancestor group scope. The value of this

element MUST not refer to a report item within

PageSection or PageHeaderFooter element

**</Visibility>**

**<CustomProperties>** (opzionale, 1 sola volta) Collezione di proprietà custom

<CustomProperty> Proprietà custom passate per il rendering di CustomReportItem

<Name>String</Name> (obbligatorio) Identificatore della proprietà

<Value>Expression</Value> (obbligatorio) value della proprietà

</CustomProperty>

**</CustomProperties>**

**<PageBreak>** (opzionale, 1 sola volta) Interruzione di pagina

<Disabled>Boolean</Disabled> (opzionale, 1 sola volta, default=false) Specifica quando l’interruzione di pagina deve essere ignorata

<ResetPageNumber>Boolean</ResetPageNumber> (opzionale, 1 sola volta, default=false) Specifica quando la numerazione della pagina deve essere reimpostata

<BreakLocation>String\*</BreakLocation> (obbligatorio, 1 sola volta) Specifica quando deve avvenire l’interruzione di pagina

**</PageBreak>**

String\*

**None**: Specifies that the **PageBreak.BreakLocation** element is ignored.

**Start**: Specifies that there MUST be a page break before a report item or each instance of a group.

**End**: Specifies that there MUST be a page break after a report item or each instance of a group.

**StartAndEnd**: Specifies that there MUST be a page break both before and after a report item or each instance of a group.

**Between**: Specifies that there MUST be a page break between each instance of a group. This value MUST NOT apply to report items.

**<TablixHeader>** (opzionale, 1 sola volta) Header della cella

<[CellContens](#CellContents)>...</CellContens>

<Size>RdlSize</Size> (obbligatorio, 1 sola volta) Altezza o larghezza dell’Header

</CustomProperty>

**</TablixHeader>**

Codice esempio

Consiglio di vedere quelli riportati in questi link:

<https://msdn.microsoft.com/en-us/library/ee208912(v=sql.105).aspx>

<https://msdn.microsoft.com/en-us/library/ee208845(v=sql.105).aspx>

<https://msdn.microsoft.com/en-us/library/ee240480(v=sql.105).aspx>

### TextBox

Elemento opzionale che può essere inserito più di una volta.

Può essere inserito nell’header e/o nel footer.

[I metodi generali già definiti nelle sezioni precedenti non li riscrivo qui sotto]

<Textbox Name= "String">

<[Style](#Style)>... </Style> (opzionale)

  <[ActionInfo](#ActionInfo)>...</ActionInfo> (opzionale, 1 sola volta) Specifica una o più azioni associate al Textbox

  <Top>RdlSize</Top> (opzionale, 1 sola volta)

  <Left>RdlSize</Left> (opzionale, 1 sola volta)

<Height>RdlSize</Height> (opzionale, 1 sola volta) Altezza del Textbox

  <Width>RdlSize</Width> (opzionale, 1 sola volta)

  <ZIndex>Integer</ZIndex> (opzionale, 1 sola volta) specifies the drawing order of a Textbox within its container

  <[Visiblity](#Visibility)>...</Visibility> Come spiegato nella sezione delle tablix

<ToolTip>String</ToolTip> (opzionale, 1 sola volta)

< DocumentMapLabel>String</ DocumentMapLabel> (opzionale, 1 sola volta) element specifies a label to identify a Textbox within the client UI in order to provide a user-friendly label for searching

  <Bookmark>String</Bookmark> (opzionale, 1 sola volta) Segnalibro

  <RepeatWith>String</RepeatWith> (opzionale, 1 sola volta) specifies the name of a data region in which a Textbox should be repeated with if the data region spans multiple pages. The data region MUST be in the same container as the text box. If the text box is within a PageSection or PageHeaderFooter, this element is ignored.

  <[CustomProperties](#CustomProperties)>...</CustomProperties> Come spiegato nella sezione delle tablix

  <Paragraphs>

  <Paragraph> (obbligatorio) Specifica le proprietà di layout di un paragrafo di testo contenuto in un TextBox

  <[Style](#Style)>...</Style>

<TextRuns> Collezione di TextRun

  <TextRun> (obbligatorio) valore e formattazione di campi di testo contigui

  <[Style](#Style)>...</Style>

  <Value>Expression</Value> (obbligatorio, 1 sola volta) Valore da mostrare

  <Label>String</label> (opzionale, 1 sola volta) La label per il segnaposto del Textbox

  <[ActionInfo](#ActionInfo)>...</ActionInfo>

  <ToolTip>String</ToolTip> (opzionale, 1 sola volta) specifies a textual tool tip label for a TextRun

  <MarkupType>String</MarkupType> (opzionale, 1 sola volta) **None**: Specifies that no markup MUST be processed and that all markup MUST be left as is. **HTML**: Specifies that HTML markup appearing in TextRun.Value MUST be processed and displayed in supporting rendering extensions.

  </TextRun>

  </TextRuns>

  <LeftIndent>RdlSize</LeftIndent> (opzionale, 1 sola volta) Intentazione del testo contenuto in un paragrafo dal bordo sinistro del textbox

  <RightIndent>RdlSize</RightIndent> Come left

  <HangingIndent>RdlSize</HangingIndent> (opzionale, 1 sola volta) specifies the indentation of the first line of text within a Paragraph relative to the peer Paragraph.LeftIndent element of the paragraph

  <ListStyle>String</ListStyle> (opzionale, 1 sola volta) **None**: Specifies that the paragraph has no list formatting. **Numbered**: Specifies that the paragraph is part of a numbered list. **Bulleted**: Specifies that the paragraph is part of a bulleted list.

  <ListLevel>Integer</ListLevel> (opzionale, 1 sola volta) specifies the numbering style and/or indentation level of a Paragraph within a Paragraphs collection.

  <SpaceBefore>RdlSize</SpaceBefore> (opzionale, 1 sola volta) spazio verticale prima del paragrafo

  <SpaceAfter>RdlSize</SpaceAfter> (opzionale, 1 sola volta) spazio verticale dopo del paragrafo

  </Paragraph>

  </Paragraphs>

  <CanGrow>Boolean</CanGrow> (opzionale, 1 sola volta, default = false) Specifica se il textbox può crescere in altezza per far entrare il contenuto

<CanShrink>Boolean</CanShrink> (opzionale, 1 sola volta, default = false) Contrario di CanGrow

  <ToggleImage> (opzionale, 1 sola volta) Specifica se l’immagine deve essere mostrata

<InitialState>Boolean</InitialState> If ToggleImage.InitialState is set to true, the toggle image is interpreted as expanded and MUST show a minus sign. If ToggleImage.InitialState is set to false, the toggle image is interpreted as collapsed and MUST show a plus sign.

</ToggleImage>

  <KeepTogether>Boolean</KeepTogether> (opzionale, 1 sola volta, default = false) Il render prova a far entrare l’intero texbox in una pagina

  <DataElementName>String</DataElementName> (opzionale, 1 sola volta) Specifica il nome da usare per il data element o l’attributo del textbox

<DataElementOutput>String</DataElementOutput> (opzionale, 1 sola volta) specifies whether a Textbox should appear in a data (XML, CSV) rendering. The value of this element MUST be one of the following: **Auto**: Specifies the default setting for how the text box will appear in a data rendering, which MUST be the same as "NoOutput" if Visibility.Hidden for the text box is set to true and Visibility.ToggleItem is not set. Otherwise, this value is interpreted as "Output". **Output**: Specifies that the text box appears in a data rendering output. **NoOutput**: Specifies that the text box does not appear in a data rendering output.

<DataElementStyle>String</DataElementStyle> (opzionale, 1 sola volta) specifies whether all TextRun.Value elements or all Textbox.Value elements, when rendered out to an XML data renderer, are rendered as attributes or as elements. The String MUST be one of the following: **Auto**(default): Specifies that this value MUST be the same as Report.DataElementStyle. **Attribute**: Specifies that TextRun.Value elements or the Textbox.Value element within this text box MUST all be rendered as attributes. **Element**: Specifies that TextRun.Value elements or the TextBox.Value element within this text box MUST all be rendered as elements.

</Textbox>

**<ActionInfo>** (opzionale, 1 sola volta) Specifica una o più azioni associate ad un item

  <Actions> (opzionale, 1 sola volta) Collezione di azioni associate ad un item

  <Action> Specifica una azione hyperlink, bookmark link, or drillthrough per un item. This element MUST be specified at least once within an Actions collection.

  <Hyperlink>String</Hyperlink> Link da visualizzare es. http://www.temposrl.it

  <BookmarkLink>String</BookmarkLink> Segnalibro da visualizzare. Mettere id del segnalibro

<Drillthrough> Specifies properties for a Report that SHOULD be executed by clicking the containing report item that is associated with an Action.Drillthrough element.

<ReportName>String</ReportName> Location del report da visualizzare. The location MUST specify, as a non-empty string (ignoring white space), either the absolute path (such as "/salesreports/orderdetails") or the relative path (such as "orderdetails") to a report on the same server. Relative paths start in the same folder as the main report.

  <Parameters> (se presente deve avere almeno 1 parametro) Parametri da passare al report

  <Parameter>

  <Omit>Boolean</Omit> (opzionale, 1 sola volta, default = false) Specifica se ignorare il parametro

  <Value>Expression</Value> (opzionale, 1 sola volta) valori da passare in input al report

<Parameter>

  </Parameters>

</Drillthrough>

</Action>

  </Actions>

**</ActionInfo>**

Codice esempio

Consiglio di vedere quello riportato in questo link:

<https://msdn.microsoft.com/en-us/library/ee223980(v=sql.105).aspx>

### Rectangle

Elemento opzionale che può essere usato come contenitore

[I metodi generali già definiti nelle sezioni precedenti non li riscrivo qui sotto]

**<Rectangle Name=** “String”**>**

  <[Style](#Style)>...</Style> (opzionale, 1 sola volta) già definito

  <[ActionInfo](#ActionInfo)>...</ActionInfo> (opzionale, 1 sola volta) già definito

  <Top>RdlSize</Top> (opzionale, 1 sola volta)

  <Left>RdlSize</Left> (opzionale, 1 sola volta)

  <Height>RdlSize</Height> (opzionale, 1 sola volta)

  <Width>RdlSize</Width> (opzionale, 1 sola volta)

  <ZIndex>Integer</ZIndex> (opzionale, 1 sola volta) specifies the drawing order of a Rectangle within its container

  <[Visibility](#Visibility)>...</Visibility> (opzionale, 1 sola volta) già definito

  <ToolTip>String</ToolTip> (opzionale, 1 sola volta) specifies the tool tip text for a Rectangle

  <DocumentMapLabel>String</DocumentMapLabel> (opzionale, 1 sola volta) specifies a label to identify a Rectangle within the client UI in order to provide a user-friendly label for searching

  <LinkToChild>String</LinkToChild> (opzionale, 1 sola volta) specifies the name of a report item that is contained within a Rectangle that is the target location for the peer DocumentMapLabel element. The LinkToChild element is ignored if the peer DocumentMapLabel element is not present.

  <Bookmark>String</Bookmark> (opzionale, 1 sola volta) segnalibro

  <RepeatWith>String</RepeatWith> (opzionale, 1 sola volta) specifies the name of a data region in which a Rectangle should be repeated if that data region spans multiple pages. The data region MUST be in the same ReportItems collection as the Line. If this rectangle is within a PageSection or PageHeaderFooter, the Rectangle.RepeatWith element is ignored.

  <[CustomProperties](#CustomProperties)>...</CustomProperties> (opzionale, 1 sola volta) già definito

  <[ReportItems](#ReportItems)>...</ReportItems> (opzionale, 1 sola volta) già definito

  <[PageBreak](#PageBreak)>...</PageBreak> (opzionale, 1 sola volta) Interruzione di pagina

  <PageName>String</PageName> (opzionale, 1 sola volta) specifies the value to use for the name of a paginated page

  <KeepTogether>Boolen</KeepTogether> (opzionale, 1 sola volta, default = false) Mantiene l’intero contenuto del rettangolo in una pagina se possibile

  <OmitBorderOnPageBreak>Boolen</OmitBorderOnPageBreak> (opzionale, 1 sola volta)

  <DataElementName>String</DataElementName> (opzionale, 1 sola volta) specifies the name to use for the data element or the data attribute of a Rectangle

  <DataElementOutput>String</DataElementOutput> (opzionale, 1 sola volta) **Auto**: Specifies the default setting for how a rectangle appears in a data rendering, which is the same as "NoOutput" if the value of the Visibility.Hidden property for the rectangle is set to true and the Visibility.ToggleItem property is not specified. Otherwise, this value is the same as "ContentsOnly". **Output**: Specifies that the rectangle appears in a data rendering output. **NoOutput**: Specifies that the rectangle does not appear in a data rendering output. **ContentsOnly**: Specifies that the rectangle does not appear in a data rendering output, but that its inner contents appear in a data rendering output.

**</Rectangle>**

Codice esempio

Consiglio di vedere quello riportato in questo link:

<https://msdn.microsoft.com/en-us/library/ee209940(v=sql.105).aspx>

Codice esempio

 <Rectangle Name="Rectangle1">

  <ReportItems>

QUALSIASI REPORT ITEM

  </ReportItems>

  <KeepTogether>true</KeepTogether>

  <Top>0.56125in</Top>

  <Left>1.11333in</Left>

  <Height>1.94792in</Height>

  <Width>4.25in</Width>

<Style>

  <BackgroundColor>Yellow</BackgroundColor>

  <Border>

  <Style>None</Style>

  </Border>

  </Style>

 </Rectangle>

### Subreport

Elemento opzionale.

<Subreport Name="String">

  <[Style](#Style)>...</Style> (opzionale, 1 sola volta)

  <[ActionInfo](#ActionInfo)>...</ActionInfo> (opzionale, 1 sola volta)

  <[Visiblity](#Visibility)>...</Visiblity> (opzionale, 1 sola volta)

  <ToolTip>String</ToolTip> (opzionale, 1 sola volta) specifies the tooltip text for a Subreport.

  <DocumentMapLabel>String</DocumentMapLabel> (opzionale, 1 sola volta) specifies a label to identify a Subreport within the client UI in order to provide a user-friendly label for searching

  <Bookmark>String</Bookmark> (opzionale, 1 sola volta) segnalibro

  <[CustomProperties](#CustomProperties)>...</CustomProperties> (opzionale, 1 sola volta)

  <ReportName>/Path/Report.rdl o .rdlc</ReportName> (opzionale, 1 sola volta) nome e path Subreport.

<Parameters> (se presente deve avere almeno 1 parametro) Parametri da passare al report. [A mio parere è inutile se passiamo un report creato in runtime da c# essendo che passiamo i dati già filtrati…]

<Parameter>

  <Omit>Boolean</Omit> (opzionale, 1 sola volta, default = false) Specifica se ignorare il parametro

  <Value>Expression</Value> (opzionale, 1 sola volta) valori da passare in input al report

<Parameter>

</Parameters>

  <Top>RdlSize</Top> (opzionale, 1 sola volta)

  <Left>RdlSize</Left> (opzionale, 1 sola volta)

  <Height>RdlSize</Height> (opzionale, 1 sola volta)

  <Width>RdlSize</Width> (opzionale, 1 sola volta)

  <ZIndex>Integer</ZIndex> (opzionale, 1 sola volta)

</Subreport>

### Image

Elemento opzionale che può essere inserito più di una volta.

Visualizza una immagine.

<Image Name="String">

  <[Style](#Style)>...</Style> (opzionale, 1 sola volta)

  <[ActionInfo](#ActionInfo)>...</ActionInfo> (opzionale, 1 sola volta)

  <Top>RdlSize</Top> (opzionale, 1 sola volta)

  <Left>RdlSize</Left> (opzionale, 1 sola volta)

  <Height>RdlSize</Height> (opzionale, 1 sola volta)

  <Width>RdlSize</Width> (opzionale, 1 sola volta)

  <ZIndex>Integer</ZIndex> (opzionale, 1 sola volta)

  <[Visibility](#Visibility)>...</Visibility> (opzionale, 1 sola volta)

  <ToolTip>String</ToolTip> (opzionale, 1 sola volta)

  <DocumentMapLabel>String</DocumentMapLabel> (opzionale, 1 sola volta)

  <Bookmark>String</Bookmark> (opzionale, 1 sola volta)

  <RepeatWith>String</RepeatWith> (opzionale, 1 sola volta)

  <[CustomProperties](#CustomProperties)>...</CustomProperties> (opzionale, 1 sola volta)

<Source>String\*</Source> (Obbligatorio, 1 sola volta) Specifica il tipo di fonte dell’immagine.

<Value>String\*\*</Value> (Obbligatorio, 1 sola volta) Specifica o il path o i data di una immagine. Dipende dal valore assegnato a Source

  <MIMEType>String</MIMEType> (Obbligatorio solo se Source = Database altrimenti ignorato, 1 sola volta) Specifica il formato dell’immagine

  <Sizing>String</Sizing> (opzionale, 1 sola volta) **AutoSize** (default): Specifies that the border of the image instance will grow or shrink to accommodate the dimensions of the actual image. **Fit**: Specifies that the actual image is resized to exactly match both the Image.Height and Image.Width dimensions. **FitProportional**: Specifies that the actual image should resize to fit within the image instance dimensions, while preserving the aspect ratio. **Clip**: Specifies that the actual image should be clipped to fit the dimensions of the image instance.

</Image>

String\*

**External:**Specifies that the peer Image.Value element MUST contain a **String** constant or an expression that evaluates to the location of an image.

**Embedded:**Specifies that the peer **Image.Value** element MUST contain a **String** constant or an expression that evaluates to the name of an embedded image in the report.

**Database:**Specifies that peer **Image.Value** element MUST contain an expression (such as a field in the database) that evaluates to the binary data for an image.

String\*\*

If the peer **Image.Source** element is set to "External", the value of the **Image.Value** element MUST be a [String](https://msdn.microsoft.com/en-us/library/ee223804(v=sql.105).aspx) ([[XMLSCHEMA2]](http://go.microsoft.com/fwlink/?LinkId=90610) section 3.2.1) constant or an expression that evaluates to the location of an [Image](https://msdn.microsoft.com/en-us/library/ee209441(v=sql.105).aspx). This location MUST be a [ReportPath](https://msdn.microsoft.com/en-us/library/ee208835(v=sql.105).aspx) or an [RdlURL](https://msdn.microsoft.com/en-us/library/ee224023(v=sql.105).aspx) value.

If the peer **Image.Source** element is set to "Embedded", the value of the **Image.Value** element MUST be a **String** constant or an expression that evaluates to the name of an [EmbeddedImage](https://msdn.microsoft.com/en-us/library/ee224069(v=sql.105).aspx) in the report.

If the peer **Image.Source** element is set to "Database", the value of the **Image.Value** element MUST be a **String** constant or an expression that evaluates to the binary data for an image.

If the **Image.Value** element has an empty value, an image MUST NOT be displayed.

Codice esempio

  <Image Name="Image1">

  <Source>External</Source>

  <Value>http://www.temposrl.it/images/logo1.png</Value>

  <Sizing>FitProportional</Sizing>

  <Height>1.26042in</Height>

  <Width>4.25in</Width>

  <Style>

  <Border>

  <Style>None</Style>

  </Border>

  <PaddingLeft>1pt</PaddingLeft>

  <PaddingRight>1pt</PaddingRight>

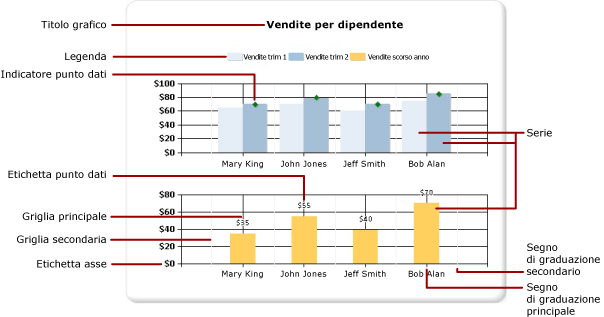
  <PaddingTop>1pt</PaddingTop>

  <PaddingBottom>1pt</PaddingBottom>

  </Style>

  </Image>

### Chart



Il grafico dei reporting services è possibile confrontarlo con le matrici in quanto dal punto di vista concettuale l’organizzazione è identica:

* Il gruppo di colonne nella matrice è area gruppi di categorie nel grafico.
* Il gruppo di righe nella matrice è area gruppi di serie nel grafico.
* L'area dati della matrice è nell'area valori nel grafico.
* Il grafico come la matrice è associato ad un unico set di dati

I grafici forniscono una visualizzazione di riepilogo dei dati. Tuttavia, nel caso di set di dati di grandi dimensioni, le informazioni di un grafico possono risultare nascoste o illeggibili. I punti dati mancanti o Null, i tipi di dati non appropriati per il tipo di grafico e applicazioni avanzate quale la combinazione di grafici con tabelle possono influire sulla leggibilità di un grafico. Prima di progettare un grafico, è opportuno preparare e identificare con attenzione i dati, in modo da procedere rapidamente e con maggiore efficienza.

I grafici risultano più efficaci quando il numero di elementi visualizzati presenta un'immagine chiara delle informazioni sottostanti per cui è possibile consolidare i dati i un grafico in molti modi:

* Quando si usa un grafico a torta, raccogliere le sezioni piccole in un'unica sezione denominata "Altro". In questo modo si riduce il numero di sezioni del grafico a torta.
* Evitare di usare le etichette dei data points quando sono presenti numerosi data points. Tali etichette sono più efficaci quando il grafico include solo pochi punti dati.
* Filtrare i dati indesiderati o irrilevanti. Per impostazione predefinita, il grafico usa la funzione predefinita Sum per aggregare valori che fanno parte dello stesso gruppo in un singolo data point nella serie. Se si modifica la funzione di aggregazione di una serie, è necessario modificarla anche nell'espressione di filtro.
* Per visualizzare dati proporzionali in un modello di matrice, usare un misuratore lineare anziché un grafico a barre. I misuratori sono più indicati per la visualizzazione di un singolo valore in una cella.

<Chart Name="String">

         <[Style](#Style)>...</Style> (opzionale, 1 sola volta)

  <Top>RdlSize</Top> (opzionale, 1 sola volta)

  <Left>RdlSize</Left> (opzionale, 1 sola volta)

  <Height>RdlSize</Height> (opzionale, 1 sola volta)

  <Width>RdlSize</Width> (opzionale, 1 sola volta)

  <ZIndex>Integer</ZIndex> (opzionale, 1 sola volta)

  <[Visibility](#Visibility)>...</Visibility> (opzionale, 1 sola volta)

  <ToolTip>String</ToolTip> (opzionale, 1 sola volta)

  <DocumentMapLabel>String</DocumentMapLabel> (opzionale, 1 sola volta)

  <Bookmark>String</Bookmark> (opzionale, 1 sola volta)

  <[CustomProperties](#CustomProperties)>...</CustomProperties> (opzionale, 1 sola volta)

         <DataSetName>String</DataSetName> (opzionale, 1 sola volta) Il nome del Dataset associato al grafico. È obbligatorio definirlo se: il grafico non contiene altri data region or se il report contiene più di un Dataset.

         <[PageBreak](#PageBreak)>...</PageBreak> (opzionale, 1 sola volta)

  <PageName>String</PageName> (opzionale, 1 sola volta) Specifica il nome della pagina paginata.

  <[Filters](#Filters)>...</Filters> (opzionale, 1 sola volta)

         <ChartCategoryHierarchy> (Obbligatorio, 1 sola volta) Specifica la gerarchia delle categorie in un grafico

           <[ChartMembers](#ChartMembers)>...</ChartMembers> (Obbligatorio, 1 sola volta)

         </ChartCategoryHierarchy>

         <ChartSeriesHierarchy> (Obbligatorio, 1 sola volta) Specifica la gerarchia delle serie in un grafico

           <[ChartMembers](#ChartMembers)>...</ChartMembers> (Obbligatorio, 1 sola volta)

         </ChartSeriesHierarchy>

         <[ChartData](#ChartData)>...</ChartData>

         <[ChartAreas](#ChartAreas)>...</ChartAreas>

         <[ChartLegends](#ChartLegends)>...</ChartLegends>

         <ChartTitles>

           <ChartTitle Name="String"> (Obbligatorio, 1 sola volta) Specifica le proprietà del ChartTitle

             <Caption>String</Caption> (Obbligatorio, 1 sola volta) Didascalia del titolo

  <Hidden>Boolean</Hidden> (opzionale, default=false, 1 sola volta)

             <[Style](#Style)>...</Style>

  <Position>RdlSize</Position> (opzionale, 1 sola volta)

<DockToChartArea>String</DockToChartArea> (opzionale, false, 1 sola volta) Specifica il nome dell’area del grafico dove mostrre il titolo del grafico

<DockOutsideChartArea>Boolean</DockOutsideChartArea> (opzionale, default=false, 1 sola volta) Specifica se il titolo deve essere tagliato oltre l’area del grafico

<DockOffset>Integer</DockOffset> (opzionale, default=0, false, 1 sola volta) Specifies the offset from the dock location as a percentage of the size of the Chart

<[ChartElementPosition](#ChartElementPosition)>...</ChartElementPosition> (opzionale, 1 sola volta) Specifica una posizione custom per il titolo

  <ToolTip>String</ToolTip> (opzionale, 1 sola volta)

  <[ActionInfo](#ActionInfo)>...</ActionInfo> (opzionale, 1 sola volta)

  <TextOrientation>String</TextOrientation> (opzionale, 1 sola volta) **Auto** (default): Specifies whether the orientation MUST be selected automatically based on context (for example, Rotated270 for titles that are docked on the left). **Horizontal**: Specifies horizontal text. **Rotated90**: Specifies vertical text, rotated 90 degrees. **Rotated270**: Specifies vertical text, rotated 270 degrees. **Stacked**: Specifies vertical text with no character rotation.

           </ChartTitle>

         </ChartTitles>

         <DynamicHeight>RdlSize</DynamicHeight> (opzionale, default=0, 1 sola volta) Specifica l’altezza entro cui il grafico può ridursi o ingrandirsi

         <DynamicWidth>RdlSize</DynamicWidth> (opzionale, default=0, 1 sola volta) Specifica la larghezza entro cui il grafico può ridursi o ingrandirsi

         <Palette>String\*</Palette> (opzionale, default=default, 1 sola volta)

         <PaletteHatchBehavior>String</PaletteHatchBehavior> (opzionale, default=default, 1 sola volta) **None**: Specifies that no hatching is added to data points. **Default** (default): Treated as "None". **Always**: Specifies that automatic hatching will be applied to all data points (unless Style.BackgroundHatchType is specified in ChartDataPoint.Style)

         <DataElementName>String</DataElementName> (opzionale, 1 sola volta) Specifica il nome da usare per il data element l’attributo per il grafico

         <DataElementOutput>String</DataElementOutput> (opzionale, 1 sola volta) **Output**: Specifies that the item appears in the data rendering output. **NoOutput**: Specifies that the item does not appear in the data rendering output. **Auto** (default): Specifies that the item does not appear in the data rendering output if the value of the grandchild Visibility.Hidden element of the Chart element is present and is set to true. Otherwise, the value is interpreted as "Output".

         <ChartBorderSkin> (opzionale, 1 sola volta) Specifica il bordo del grafico

           <ChartBorderSkinType>String\*\*</ChartBorderSkinType> (opzionale, 1 sola volta)

           <[Style](#Style)>...</Style> (opzionale, 1 sola volta)

         </ChartBorderSkin>

         <ChartNoDataMessage Name="String"> (Opzionale, 1 sola volta) messaggio da mostrare se il grafico non contiene dati

             <Caption>String</Caption> (Obbligatorio, 1 sola volta) Didascalia del messaggio

  <Hidden>Boolean</Hidden> (opzionale, default=false, 1 sola volta)

             <[Style](#Style)>...</Style>

  <Position>RdlSize</Position> (opzionale, 1 sola volta)

<DockToChartArea>String</DockToChartArea> (opzionale, false, 1 sola volta) Specifica il nome dell’area del grafico dove mostare il messaggio del grafico

<DockOutsideChartArea>Boolean</DockOutsideChartArea> (opzionale, default=false, 1 sola volta) Specifica se il messaggio deve essere tagliato oltre l’area del grafico

<DockOffset>Integer</DockOffset> (opzionale, default=0, false, 1 sola volta) Specifies the offset from the dock location as a percentage of the size of the Chart

<[ChartElementPosition](#ChartElementPosition)>...</ChartElementPosition> (opzionale, 1 sola volta) Specifica una posizione custom per il messaggio

  <ToolTip>String</ToolTip> (opzionale, 1 sola volta)

  <[ActionInfo](#ActionInfo)>...</ActionInfo> (opzionale, 1 sola volta)

  <TextOrientation>String</TextOrientation> (opzionale, 1 sola volta) **Auto** (default): Specifies whether the orientation MUST be selected automatically based on context (for example, Rotated270 for titles that are docked on the left). **Horizontal**: Specifies horizontal text. **Rotated90**: Specifies vertical text, rotated 90 degrees. **Rotated270**: Specifies vertical text, rotated 270 degrees. **Stacked**: Specifies vertical text with no character rotation.

         </ChartNoDataMessage>

</Chart>

String\*

* **Default**: Specifies that the Default palette should be used.
* **Berry**: Specifies that the Berry palette should be used.
* **BrightPastel**: Specifies that the BrightPastel palette should be used.
* **Chocolate**: Specifies that the Chocolate palette should be used.
* **EarthTones**: Specifies that the EarthTones palette should be used.
* **Excel**: Specifies that the Excel palette should be used.
* **Fire**: Specifies that the Fire palette should be used.
* **GrayScale**: Specifies that the GrayScale palette should be used.
* **Light**: Specifies that the Light palette should be used.
* **Pacific**: Specifies that the Pacific palette should be used.
* **PacificLight**: Specifies that the PacificLight palette should be used.
* **PacificSemiTransparent**: Specifies that the PacificSemiTransparent palette should be used.
* **Pastel**: Specifies that the Pastel palette should be used.
* **SemiTransparent**: Specifies that the SemiTransparent palette should be used.
* **SeaGreen**: Specifies that the SeaGreen palette should be used.
* **Custom**: Specifies that the Custom palette should be used.

String\*\*

* **None**: Specifies no border skin.
* **Emboss**: Specifies an embossed border skin.
* **Raised**: Specifies a raised border skin.
* **Sunken**: Specifies a sunken border skin.
* **FrameThin1**: Specifies the FrameThin1 border skin.
* **FrameThin2**: Specifies the FrameThin2 border skin.
* **FrameThin3**: Specifies the FrameThin3 border skin.
* **FrameThin4**: Specifies the FrameThin4 border skin.
* **FrameThin5**: Specifies the FrameThin5 border skin.
* **FrameThin6**: Specifies the FrameThin6 border skin.
* **FrameTitle1**: Specifies the FrameTitle1 border skin.
* **FrameTitle2**: Specifies the FrameTitle2 border skin.
* **FrameTitle3**: Specifies the FrameTitle3 border skin.
* **FrameTitle4**: Specifies the FrameTitle4 border skin.
* **FrameTitle5**: Specifies the FrameTitle5 border skin.
* **FrameTitle6**: Specifies the FrameTitle6 border skin.
* **FrameTitle7**: Specifies the FrameTitle7 border skin.
* **FrameTitle8**: Specifies the FrameTitle8 border skin.

**<ChartMembers>** (Obbligatorio, 1 sola volta) Collezione di ChartMember

<ChartMember> (Obbligatorio) Specifica le categorie o le serie di un grafico

               <[Group](#Group)>...</Group> (opzionale, 1 sola volta)

  <SortExpressions> (opzionale, 1 sola volta)

  <SortExpression> (Obbligatorio) Espressione di ordinamento

  <Value>Expression</Value>

  </SortExpression>

  </SortExpressions>

               <[ChartMembers](#ChartMembers)>...</ChartMembers> (opzionale, 1 sola volta)

               <Label>Expression\*</Label> (Obbligatorio, 1 sola volta) Label da mostrare nella leggenda

               <[CustomProperties](#CustomProperties)>...</CustomProperties> (opzionale, 1 sola volta)

               <DataElementName>String</DataElementName> (opzionale, 1 sola volta) Specifica il nome da usare per il data element per il chart member

               <DataElementOutput>String\*</DataElementOutput> (opzionale, 1 sola volta) Specifica se l’oggetto deve apparire nel data rendering.

</ChartMember>

**</ChartMembers>**

Expression\*

* If the value of the [Type](https://msdn.microsoft.com/en-us/library/ee209938(v=sql.105).aspx) element not is "Shape", the **Label** element specifies the label to display in the legend for [ChartMembers](https://msdn.microsoft.com/en-us/library/ee240019(v=sql.105).aspx) in the [ChartSeriesHierarchy](https://msdn.microsoft.com/en-us/library/ee224503(v=sql.105).aspx), and it specifies the labels to display on the **CategoryAxis** for **ChartMembers** in the [ChartCategoryHierarchy](https://msdn.microsoft.com/en-us/library/ee210028(v=sql.105).aspx).
* If the value of the **Type** element is "Shape", the **Label** element specifies the label to display in the legend for **ChartMembers** in the **ChartCategoryHierarchy**.

String\*

**Output:** Specifies that the item appears in the data rendering output.

**NoOutput:** Specifies that the item does not appear in the output.

**Auto:** If the ChartMember.Group element is present, the value of this element is interpreted as "Output".

**<ChartData>** (opzionale, 1 sola volta) Descrive la struttura dei dati in un Grafico

           <ChartSeriesCollection> (Obbligatorio, 1 sola volta) Specifica i data point per ogni serie nel grafico. Deve contenere almeno 1 ChartSeries

             <[ChartSeries](#ChartSeries)>...</ChartSeries> (Obbligatorio, Identificatore) Specifica la lista dei data points per una serie nella ChartArea.

           </ChartSeriesCollection>

**</ChartData>**

**<ChartSeries Name="String">** (Obbligatorio, Identificatore) Specifica la lista dei data points per una serie nella ChartArea.

<Hidden>Boolean</Hidden> (opzionale, 1 sola volta, default=false) Specifica se il ChartSeries deve essere nascosto

<ChartDataPoints> (Obbligatorio, ignorato se ChartSeries in ChartDerivedSeries,1 sola volta) Specifica i data point della serie. Collezione di ChartDataPoint

<ChartDataPoint> (Obbligatorio) Specifica un data point nel grafico

<ChartDataPointValues> (opzionale) Specifica un set di valori per il data point del grafico

                     <X>Expression</X> (opzionale, default=0, obbligatorio se [Type=Scatter], 1 sola volta) Specifica il valore X. L’espressione deve restituire un valore Numerico.

                     <Y>Expression</Y> (opzionale, default=0, obbligatorio se [Type!=Range] or [Type=Range and Subtype=ErrorBar], 1 sola volta) Specifica il valore Y. L’espressione deve restituire un valore Numerico.

                     <Size>Expression</Size> (opzionale, default=0, obbligatorio se [Type=Scatter and Subtype=Bubble], 1 sola volta) Specifica la grandezza. L’espressione deve restituire un valore Numerico.

                     <High>Expression</High> (non mettere se [Type=Range and Y non specificato], obbligatorio, 1 sola volta) Specifica il valore alto. L’espressione deve restituire un valore Numerico.

                     <Low>Expression</Low> (opzionale, default=0, non mettere se [Type=Range], 1 sola volta) Specifica il valore basso. L’espressione deve restituire un valore Numerico.

                     <Start>Expression</Start> (opzionale, default=0, obbligatorio se [Type=Range] AND [Subtype=Stock or =Candlestick or =BoxPlot], 1 sola volta) Specifica il valore iniziale o di apertura. L’espressione deve restituire un valore Numerico.

                     <End>Expression</End> (opzionale, obbligatorio se [Type=Range] AND [Subtype=Stock or =Candlestick or =BoxPlot], 1 sola volta) Specifica il valore finale o di chiusura. L’espressione deve restituire un valore Numerico.

                     <Mean>Expression</Mean> (opzionale, obbligatorio se [Type=Range AND Subtype=BoxPlot], 1 sola volta) Specifica il valore di mezzo. L’espressione deve restituire un valore Numerico.

                     <Median>Expression</Median> (opzionale, default=0, obbligatorio se [Type=Range AND Subtype=BoxPlot], 1 sola volta) Specifica il valore mediano. L’espressione deve restituire un valore Numerico.

</ChartDataPointValues>

<[ChartDataLabel](#ChartDataLabel)>...</ChartDataLabel> (opzionale, 1 sola volta) Specifica i data label da mostrare sui data values

<AxisLable>String</AxisLabel> (opzionale, default=“”, 1 sola volta) Specifica il label da usare sulle assi per il data point

<ToolTip>String</ToolTip> (opzionale, default=“”, 1 sola volta) Specifica il tooltip da mostrare per il data point

<[Style](#Style)>...</Style> (opzionale, 1 sola volta)

<[ActionInfo](#ActionInfo)>...</ActionInfo> (opzionale, 1 sola volta)

                   <[ChartMarker](#ChartMarker)>...</ChartMarker> (opzionale, 1 sola volta) Specifica l’aspetto di un data point marker

                   <[ChartItemInLegend](#ChartItemInLegend)>...</ChartItemInLegend> (opzionale, non mettere se [Type!=Shape], 1 sola volta) Specifica come appariranno i data point quando saranno visualizzati nella legenda

<[CustomProperties](#CustomProperties)>...</CustomProperties> (opzionale, 1 sola volta)

<DataElementName>String</DataElementName> (opzionale, 1 sola volta, identificatore) Specifica il nome da usare per il data element o l’attributo per il data point

<DataElementOutput>String</DataElementOutput> (opzionale, 1 sola volta) Specifica se l’item deve apparire nel data rendering. **Output**: Specifies that the data point appears in the data rendering output. **NoOutput**: Specifies that the data point does not appear in the data rendering output. **ContentsOnly** (default): Specifies that the data point does not appear in the output, but that its values appears in the data renderer output as if they were in the data point’s container element.

<ChartDataPoint>

<ChartDataPoints>

<Type>String\*</Type> (opzionale, 1 sola volta) Specifica il tipo di visualizzazione

<SubType>String\*\*</SubType> (opzionale, 1 sola volta) Specifica il sottotipo di visualizzazione

            <[Style](#Style)>...</Style> (opzionale, 1 sola volta)

            <ChartEmptyPoints> (opzionale, 1 sola volta) Specifica il comportamento dei data point vuoti

<[Style](#Style)>...</Style> (opzionale, 1 sola volta)

<[ChartMarker](#ChartMarker)>...</ChartMarker> (opzionale, 1 sola volta)

<[ChartDataLabel](#ChartDataLabel)>...</ChartDataLabel> (opzionale, 1 sola volta)

<AxisLabel>String</AxisLabel> (opzionale, default=“”, 1 sola volta) Specifica il label da usare sulle assi di un data point vuoto

<ToolTip>String</ToolTip> (opzionale, default=“”, 1 sola volta) Specifica il tooltip da mostrare per un data point vuoto

<[ActionInfo](#ActionInfo)>...</ActionInfo> (opzionale, 1 sola volta)

<[CustomProperties](#CustomProperties)>...</CustomProperties> (opzionale, 1 sola volta)

</ChartEmptyPoints>

            <[CustomProperties](#CustomProperties)>...</CustomProperties> (opzionale, 1 sola volta)

            <LegendName>String</LegendName> (opzionale, 1 sola volta) Specifica il nome del ChartLegend dove il ChartSeries deve apparire

            <[ChartItemInLegend](#ChartItemInLegend)>...</ChartItemInLegend> (opzionale, 1 sola volta)

            <ChartAreaName>String</ChartAreaName> (opzionale, nome della ChartArea nel Chart, 1 sola volta) Specifica il nome della CharArea nella quale disegnare il ChartSeries

            <ValueAxisName>String</ValueAxisName> (opzionale, nome del ChartAxis nel ChartArea, 1 sola volta) Specifica il nome del value axis sul quale disegnare il ChartSeries

            <CategoryAxisName>String</CategoryAxisName> (opzionale, nome del ChartAxis nel ChartCategoryAxes, 1 sola volta) Specifica il nome del gategory axis sul quale disegnare il ChartSeries

            <ChartSmartLabel> (Obbligatorio, 1 sola volta) Specifica le proprietà del label intelligente

                 <Disabled>Boolean</Disabled> (opzionale, default=false, 1 sola volta) Specifica se disabilitare lo smart label

                 <AllowOutSidePlotArea>String</AllowOutSidePlotArea> (opzionale, 1 sola volta) Specifica quali data point labels possono essere disegnati fuori dal plot area. **Partial** (default): Specifies that labels are allowed to be partially outside the plot area. **True**: Specifies that labels are allowed to be entirely outside the plot area. **False**: Specifies that labels MUST be entirely inside the plot area.

                 <CalloutBackColor>String</CalloutBackColor> (opzionale, default=empty, 1 sola volta) Specifica il colore del box che contiene il testo della label quando CalloutStyle=Box

                 <CalloutLineAnchor>String</CalloutLineAnchor> (opzionale, 1 sola volta) Specifica la forma da disegnare all'estremità di una calloutline. **None**: Specifies that the shape MUST NOT appear on the point end. **Arrow** (default): Specifies an arrow shape. **Diamond**: Specifies a diamond shape. **Square**: Specifies a square shape. **Round**: Specifies a round shape.

                 <CalloutLineColor>String</CalloutLineColor> (opzionale, Default=Black, 1 sola volta) Specifica il colore della calloutline

                 <CalloutLineStyle>String</CalloutLineStyle> (opzionale, 1 sola volta) Specifica lo stile della calloutline. **Solid** (default): Specifies a solid line. **None**: Specifies no line. **Dotted**: Specifies a dotted line. **Dashed**: Specifies a dashed line. **Double**: Specifies a double solid line. **DashDot**: Specifies a dash-dot line. **DashDotDot**: Specifies a dash-dot-dot line.

                 <CalloutLineWidth>RdlSize</CalloutLineWidth> (opzionale, default=0.75pt, 1 sola volta) Specifica lo spessore della calloutline

                 <CalloutStyle>String</CalloutStyle> (opzionale, 1 sola volta) Specifica lo stile da usare per disegnare le calloutlines. **Underline** (default): Specifies to attach the callout line to an underline on the label. **Box**: Specifies to attach the callout line to a box around the label. **None**: Specifies no additional label style for the callout line.

                 <ShowOverlapped>Boolean</ShowOverlapped> (opzionale, default=false, 1 sola volta) Specifica se le labels devono essere mostrate anche se c’è un problema di sovrapposizione che non può essere risolto.

                 <MarkerOverlapping>Boolean</MarkerOverlapping> (opzionale, default=false, 1 sola volta) indica se i point labels possono sovrapporsi ai marcatori

                 <MaxMovingDistance>RdlSize</MaxMovingDistance> (opzionale, default=23pt, 1 sola volta) Indica la massima distanza dal data point che le labels possono percorrere in caso di soprapposizione

                 <MinMovingDistance>RdlSize</MinMovingDistance> (opzionale, default=0pt, 1 sola volta) Indica la minima distanza dal data point che le labels possono percorrere in caso di soprapposizione

                 <ChartNoMoveDirections> (opzionale, 1 sola volta) Specifica in quale direzione non possono muioversi le labels

                 <Up>Boolean</Up> (opzionale, default false, 1 sola volta) Specifica che le labels non possono muoversi in alto

                 <Left>Boolean</Left> (opzionale, default false, 1 sola volta)

                 <Right>Boolean</Right> (opzionale, default false, 1 sola volta)

                 <Down>Boolean</Down> (opzionale, default false, 1 sola volta)

                 <UpLeft>Boolean</UpLeft> (opzionale, default false, 1 sola volta)

                 <UpRight>Boolean</UpRight> (opzionale, default false, 1 sola volta)

                 <DownLeft>Boolean</DownLeft> (opzionale, default false, 1 sola volta)

                 <DownRight>Boolean</DownRight> (opzionale, default false, 1 sola volta)

</ChartNoMoveDirections>

            </ChartSmartLabel>

            <[ChartDataLabel](#ChartDataLabel)>...</ChartDatalabel> (opzionale, 1 sola volta)

            <[ChartMarker](#ChartMarker)>...</ChartMarker> (opzionale, 1 sola volta)

**</ChartSeries>**

String\*

* **Column**: Specifies a column chart.
* **Bar**: Specifies a bar chart.
* **Line**: Specifies a line chart.
* **Shape**: Specifies a shape chart.
* **Scatter**: Specifies a scatter chart.
* **Area**: Specifies an area chart.
* **Range**: Specifies a range chart.
* **Polar**: Specifies a polar chart.

String\*\*

* **Plain**: Specifies a plain visualization for all types.
* **Stacked**: Specifies a stacked visualization for column, bar, and area charts. This value SHOULD NOT be used if the value of the **ChartSeries.Type** element is not "Column", "Bar", or "Area".
* **PercentStacked**: Specifies a column, bar, or area chart stacked by percentage. This value SHOULD NOT be used if the value of the **ChartSeries.Type** element is not "Column", "Bar", or "Area".
* **Smooth**: Specifies a smooth line, area, or range chart. This value SHOULD NOT be used if the value of the **ChartSeries.Type** element is not "Line", "Area", or "Range".
* **Stepped**: Specifies a stepped line chart. This value SHOULD NOT be used if the value of the **ChartSeries.Type** element is not "Line".
* **Pie**: Specifies a pie chart. This value SHOULD NOT be used if the value of the **ChartSeries.Type** element is not "Shape".
* **ExplodedPie**: Specifies an exploded pie chart. This value SHOULD NOT be used if the value of the **ChartSeries.Type** element is not "Shape".
* **Doughnut**: Specifies a doughnut chart. This value SHOULD NOT be used if the value of the **ChartSeries.Type** element is not "Shape".
* **Exploded Doughnut**: Specifies an exploded doughnut chart. This value SHOULD NOT be used if the value of the **ChartSeries.Type** element is not "Shape".
* **Funnel**: Specifies a funnel chart. This value SHOULD NOT be used if the value of the **ChartSeries.Type** element is not "Shape".
* **Pyramid**: Specifies a pyramid chart. This value SHOULD NOT be used if the value of the **ChartSeries.Type** element is not "Shape".
* **Bubble**: Specifies a bubble chart. This value SHOULD NOT be used if the value of the **ChartSeries.Type** element is not "Scatter".
* **Candlestick**: Specifies a candlestick chart. This value SHOULD NOT be used if the value of the **ChartSeries.Type** element is not "Range".
* **Stock**: Specifies a stock chart. This value SHOULD NOT be used if the value of the **ChartSeries.Type** element is not "Range".
* **Bar**: Specifies a range bar chart. This value SHOULD NOT be used if the value of the **ChartSeries.Type** element is not "Range".
* **Column**: Specifies a range column chart. This value SHOULD NOT be used if the value of the **ChartSeries.Type** element is not "Range".
* **BoxPlot**: Specifies a box plot chart. This value SHOULD NOT be used if the value of the **ChartSeries.Type** element is not "Range".
* **ErrorBar**: Specifies an error bar chart. This value SHOULD NOT be used if the value of the **ChartSeries.Type** element is not "Range".
* **Radar**: Specifies a radar chart. This value SHOULD NOT be used if the value of the **ChartSeries.Type** element is not "Polar".
* **TreeMap**: In RDL schema 2016/01, specifies a TreeMap chart. This value SHOULD NOT be used if the value of the **ChartSeries.Type** element is not "Shape".
* **Sunburst**: In RDL schema 2016/01, specifies a Sunburst chart. This value SHOULD NOT be used if the value of the **ChartSeries.Type** element is not "Shape".
* If the **ChartSeries.Subtype** element is not present and if the value of the **ChartSeries.Type** element is "Shape", the value of the **ChartSeries.Subtype** element is interpreted as "Pie".
* If the **ChartSeries.Subtype** element is not present and if the value of the **ChartSeries.Type** element is not "Shape", the value of the **ChartSeries.Subtype** element is interpreted as "Plain".

String\*\*\*

* **Auto**: Specifies automatic positioning.
* **Top**: Specifies to position the label at the top of the data point.
* **TopLeft**: Specifies to position the label at the top-left of the data point.
* **TopRight**: Specifies to position the label at the top-right of the data point.
* **Left**: Specifies to position the label to the left of the data point.
* **Center**: Specifies to position the label at the center of the data point.
* **Right**: Specifies to position the label at the right of the data point.
* **BottomRight**: Specifies to position the label at the bottom-right of the data point.
* **Bottom**: Specifies to position the label at the bottom of the data point.
* **BottomLeft**: Specifies to position the label at the bottom-left of the data point.
* **Outside**: Specifies to position the label outside of the data point. If the subtype of the containing series, as specified in [ChartSeries.Subtype](https://msdn.microsoft.com/en-us/library/ee239962(v=sql.105).aspx), is not "Pie" or "Doughnut", "Outside" MUST be treated as "Top".

String°

* **None**: Specifies that the marker MUST not be displayed.
* **Square**: Specifies a square marker.
* **Circle**: Specifies a circle marker.
* **Diamond**: Specifies a diamond marker.
* **Triangle**: Specifies a triangle marker.
* **Cross**: Specifies a cross marker.
* **Star4**: Specifies a four-pointed star marker.
* **Star5**: Specifies a five-pointed star marker.
* **Star6**: Specifies a six-pointed star marker.
* **Star10**: Specifies a ten-pointed star marker.
* **Auto**: Specifies to automatically cycle through marker types for each series.

**<ChartDataLabel>** (opzionale, 1 sola volta) Specifica i data label da mostrare sui data values

<Visibile>Boolean</Visible> (opzionale, 1 sola volta, default=false) specifica se il DataLabel è visibile

<[Style](#Style)>...</Style> (opzionale, 1 sola volta)

<Label>String</Label> (opzionale, non mettere se [UseValueAsLabel=true], default=“”, 1 sola volta) Testo del DataLabel

<UseValueAsLabel>Boolean</UseValueAsLabel> (opzionale, default=false,1 sola volta) Specifica che se Y è presente deve essere usato come data label

<Position>String\*\*\*</Position> (opzionale, 1 sola volta) Specifica la posizione del label per il data point

<Rotation>Integer</Rotation> (opzionale, default=0, 1 sola volta) Specifica l’angolo di rotazione del data label

<ToolTip>String</ToolTip> (opzionale, default=“”, 1 sola volta) Specifica il tooltip da mostrare per il data label

<[ActionInfo](#ActionInfo)>...</ActionInfo> (opzionale, 1 sola volta)

**</ChartDataLabel>**

**<ChartMarker>** (opzionale, 1 sola volta) Specifica l’aspetto di un data point marker

<Type>String°</Type> (opzionale, 1 sola volta) Tipo del marcatore

<Size>RdlSize</Size> (opzionale, default=3.75pt, 1 sola volta) Specifica la grandezza del marcatore

<[Style](#Style)>...</Style> (opzionale, 1 sola volta)

**</ChartMarker>**

**<ChartItemInLegend>** (opzionale, non mettere se [Type!=Shape], 1 sola volta) Specifica come appariranno i data point quando saranno visualizzati nella legenda

<LegendText>String</LegendText> (opzionale, 1 sola volta) Specifica il label da usare nella legenda. **If** the ChartItemInLegend.LegendText element is not present and if the parent is a ChartDataPoint element, ChartMember.Label elements will be used from the Chart.ChartCategoryHierarchy element, concatenated with a hyphen character (-) between each member. **If** the ChartItemInLegend.LegendText element is not present and if the parent of the chart item is not a ChartDataPoint element, its value is interpreted as an empty string.

<ToolTip>String</ToolTip> (opzionale, 1 sola volta) Specifica il tool tip da mostrare per un oggetto nella legenda

<[ActionInfo](#ActionInfo)>...</ActionInfo> (opzionale, 1 sola volta)

<Hidden>Boolean</Hidden> (opzionale, 1 sola volta) Specifica se nascondere l’oggetto nella legenda

**</ChartItemInLegend>**

**<ChartAreas> (opzionale, 1 sola volta) Collezione di ChartArea**

<ChartArea Name="String"> **(Obbligatorio) Collezione di ChartArea**

<Hidden>Boolean</Hidden> (opzionale, default=false, 1 sola volta) Specifica se nascondere l’oggetto

<ChartCategoryAxes> (1 sola volta) Istanze di categorie sull’ asse X. The ChartCategoryAxes element SHOULD NOT contain more than one ChartAxis that has ChartAxis.Location set to "Default". Additionally, the ChartCategoryAxes element SHOULD NOT contain more than one ChartAxis that has ChartAxis.Location set to "Opposite".

<[ChartAxis](#ChartAxis)>...</ChartAxis>

</ChartCategoryAxes>

<ChartValueAxes> (1 sola volta) Lista dei valori sull’asse Y. The ChartValueAxes element SHOULD NOT contain more than one ChartAxis that has the ChartAxis.Location element set to "Default". Additionally, the ChartValueAxes element SHOULD NOT contain more than one chart axis that has the ChartAxis.Location element set to "Opposite".

<[ChartAxis](#ChartAxis)>...</ChartAxis>

</ChartValueAxes>

<ChartThreeDProperties> (opzionale, 1 sola volta) Specifica le proprietà di un layout 3D

<Enabled>Boolean</Enabled> (opzionale, default=false, 1 sola volta) Specifica se le proprietà 3D sono abilitate

<ProjectionMode>String</ProjectionMode> (opzionale, 1 sola volta) **Oblique** (default): Specifies that an oblique projection MUST be used. **Perspective**: Specifies that a perspective projection MUST be used.

<Rotation>-181<Integer<181</Rotation> (opzionale, default=30, 1 sola volta) Specifica l’angolo di rotazione

<Inclination>0≤Integer<91</Inclination> (opzionale, default=30, 1 sola volta) Specifica l’angolo di inclinazione

<Perspective>0≤Integer<101</Perspective> (opzionale, ignorato se [ProjectionMode!=Perspective], default=0, 1 sola volta) Specifica la percentuale di prospettiva

<DepthRatio>0≤Integer<1001</DepthRatio> (opzionale, default=100, 1 sola volta) Specifica il rapporto tra profondità e lagherzza in percentuale

<Shading>String</Shading> (opzionale, 1 sola volta) **Real** (default): Specifies realistic shading. **Simple**: Specifies simplified shading. **None**: Specifies no shading.

<GapDepth>0≤Integer<2147483648</GapDepth> (opzionale, default=30, 1 sola volta) specifies the depth gap between 3D bars and columns as a percentage of the distance between data points in one row

<WallThickness>0≤Integer<31</WallThickness> (opzionale, default=7, 1 sola volta) Specifica lo spessore delle pareti esterne

<Clustered>Boolean</Clustered> (opzionale, default=false, 1 sola volta) Specifica se le serie di dati sono raggruppati

</ChartThreeDProperties>

<[Style](#Style)>...</Style> (opzionale, 1 sola volta)

<AlignOrientation>String</AlignOrientation> (opzionale, 1 sola volta) **None** (default): Specifies that no alignment MUST be applied. **Vertical**: Specifies that vertical alignment MUST be applied. **Horizontal**: Specifies that horizontal alignment MUST be applied. **All**: Specifies that both vertical and horizontal alignment MUST be applied.

<ChartAlignType> (opzionale, 1 sola volta) specifies which aspects of a ChartArea SHOULD be aligned with the target chart area. If this element is present, ChartArea.AlignWithChartArea SHOULD be specified.

<AxesView>Boolean</AxesView> (opzionale, default=false, 1 sola volta) Specifica se le aree del grafico devono essere allineate sulle viste degli assi

<Cursor>Boolean</Cursor> (opzionale, default=false, 1 sola volta) Specifica se le aree del grafico devono essere alleate con i cursori

<Position>Boolean</Position> (opzionale, default=false, 1 sola volta) Specifica se le aree del grafico devono essere alleate con le posizioni delle aree del grafico

<InnerPlotPosition>Boolean</InnerPlotPosition> (opzionale, default=false, 1 sola volta) Specifica se le aree del grafico devono essere alleate con le posizioni interne di disegno

</ChartAlignType>

<[ChartElementPosition](#ChartElementPosition)>...</ChartElementPosition> (opzionale, 1 sola volta) Specifica una posizione custom per l’area del grafico

<[ChartInnerPlotPosition](#ChartElementPosition)>...</ChartInnerPlotPosition> (opzionale, 1 sola volta) Specifica una posizione custom per l’area di disegno del grafico

<AlignWithChartArea>String</AlignWithChartArea> (obbligatorio se [ChartAlignType è presente], opzionale, 1 sola volta) Specifica il nome dell’area del grafico da allineare a questa area del grafico

<EquallySizedAxesFont>Boolean</EquallySizedAxesFont> (opzionale, default=false, 1 sola volta) Specifica se la stessa grandezza di font deve essere usata su tutte le assi. If there is at least one ChartAxis.LabelsAutoFitDisabled element specified or evaluated as true either in ChartArea.ChartCategoryAxes or ChartArea.ChartValueAxes, the value of the ChartArea.EquallySizedAxesFont element is interpreted as false.

</ChartArea>

**</ChartAreas>**

**<ChartAxis Name="String">**   **(identificatore) Specifica le proprietà per le etichette, titoli, e linee sull’asse del grafico**

<Visible>String</Visible> (opzionale, default=Auto, 1 sola volta) Specifica se mostrare il ChartAxis. **Auto**: Specifies that the axis MUST be displayed if it is in use (such as if a series is plotted against it, or if it has the ChartAxis.ChartAxisTitle element specified). **True**: Specifies that the axis MUST be displayed. **False**: Specifies that the axis MUST be hidden.

<[Style](#Style)>...</Style>

<ChartAxisTitle> (opzionale, 1 sola volta) Specifica il titolo del ChartAxis

<Caption>String</Caption> (Obbligatorio, 1 sola volta) Specifica la didascalia del titolo.

<Position>String</Position> (opzionale, 1 sola volta) **Center** (default): Specifies that the axis title MUST be positioned on the center of the axis. **Near**: Specifies that the axis title MUST be positioned on the near side of the axis (the near side is the origin of the primary axes). **Far**: Specifies that the axis title MUST be positioned on the far side of the axis.

<[Style](#Style)>...</Style> (opzionale, 1 sola volta)

<TextOrientation>String</TextOrientation> (opzionale, 1 sola volta) **Horizontal**: Specifies horizontal text. **Rotated90**: Specifies vertical text, rotated 90 degrees.Rotated270: Specifies vertical text, rotated 270 degrees. **Stacked**: Specifies vertical text with no character rotation. **Auto** (default): Specifies that the orientation is selected automatically based on context (for example, "Rotated270" is selected for titles that are docked on the left).

</ChartAxisTitle>

<Margin>String</Margin> (opzionale, 1 sola volta) **Auto** (default): Specifies that the margins are included based on the ChartSeries.Type or ChartSeries.Subtype. **True**: Specifies that the axis has a margin. **False**: Specifies that the axis has no margin.

<Interval>Float</Interval> (opzionale, default=0, 1 sola volta) Specifica l’intervallo tra linee di griglia, segni di spunta e etichette del ChartAxis

<IntervalType>String\*</IntervalType> (opzionale, 1 sola volta) Specifica l’unità da usare per l’intervallo.

<IntervalOffset>Float</IntervalOffset> (opzionale, default=0, 1 sola volta) Specifica l’offset del primo segno di spunta dal minimo

<IntervalOffsetType>String\*</IntervalOffsetType> (opzionale, default=false, 1 sola volta) Specifica l’unità da usare per l’IntervalOffset.

<VariableAutoInterval>Boolean</VariableAutoInterval> (opzionale, default=false, 1 sola volta) Specifica se il valore dell’invervallo è calcolato automaticamente basandosi sulle dimensioni disponibili o se l’invervallo è calcolato solo sul data range.

<LabelInterval>Float</LabelInterval> (opzionale, default=0, 1 sola volta) Specifica l’intervallo tra le etichette del chart axis

<LabelIntervalType>String\*</LabelIntervalType> (opzionale, 1 sola volta) Specifica l’unità da usare per l’intervallo delle etichette.

<LabelIntervalOffset>Float</IntervalOffset> (opzionale, default=0, 1 sola volta) Specifica l’offset della prima etichetta dal minimo

<LabelIntervalOffsetType>String\*</IntervalOffsetType> (opzionale, default=false, 1 sola volta) Specifica l’unità da usare per l’IntervalOffset.

<ChartMajorGridLines> (opzionale, 1 sola volta) Specifica la linea di griglia maggiore

<[ChartGridLines](#ChartGridLines)>...</ChartGridLines>

</ChartMajorGridLines>

<ChartMinorGridLines> (opzionale, 1 sola volta) Specifica la linea di griglia minore

<[ChartGridLines](#ChartGridLines)>...</ChartGridLines>

</ChartMinorGridLines>

<ChartMajorTickMarks> (opzionale, 1 sola volta) Specifica il segno di spunta maggiore

<[ChartTickMarks](#ChartTickMarks)>...</ChartTickMarks>

</ChartMajorTickMarks>

<ChartMinorTickMarks> (opzionale, 1 sola volta) Specifica il segno di spunta minore

<[ChartTickMarks](#ChartTickMarks)>...</ChartTickMarks>

</ChartMinorTickMarks>

<MarksAlwaysAtPlotEdge>Boolean</MarksAlwaysAtPlotEdge> (opzionale, default=false, 1 sola volta) Specifica se le spunte devono stare nel bordo dell’area di disegno anzicché muoversi con il ChartAxis

<Reverse>Boolean</Reverse> (opzionale, default=false, 1 sola volta) Specifica se l’asse deve essere disegnate nella direzione inversa

<CrossAt>Numeric or DateTime</CrossAt> (opzionale, 1 sola volta) Specifica il valore dove si incrociano le assi

<Location>String</Location> (opzionale, 1 sola volta) **Default** (Default): Specifies that the axis MUST be drawn on the default side. **Opposite**: Specifies that the axis MUST be drawn on the opposite side.

<Interlaced>Boolean</Interlaced> (opzionale, default=false, 1 sola volta) specifies whether strip lines MUST be drawn at an interval of every other gridline for a ChartAxis. If gridlines are not used for the axis, the axis tick marks or labels are used to determine the interval of the interlaced strip lines.

<InterlacedColor>RdlColor</InterlacedColor> (opzionale, default=empty, 1 sola volta) specifies the color of interlaced strips for a ChartAxis

<ChartStripLines> (opzionale, 1 sola volta) Specifica striplines custom per l’asse del grafico

<[Style](#Style)>...</Style> (opzionale, 1 sola volta)

<Title>String</Title> (opzionale, 1 sola volta) Specifica il titolo per un stripe line

<TextOrientation>String</TextOrientation> (opzionale, 1 sola volta) **Auto** (default): Specifies whether the orientation MUST be selected automatically based on context; for example, "Rotated270" is selected for titles that are docked on the left. **Horizontal**: Specifies horizontal text. **Rotated90**: Specifies vertical text, rotated 90 degrees. **Rotated270**: Specifies vertical text, rotated 270 degrees. **Stacked**: Specifies vertical text with no character rotation.

<[ActionInfo](#ActionInfo)>...</ActionInfo> (opzionale, 1 sola volta)

<ToolTip>String</ToolTip> (opzionale, 1 sola volta)

<Interval>Float</Interval> (opzionale, default=0, 1 sola volta) Specifica l’intervallo tra gli strip lines

<IntervalType>String\*</IntervalType> (opzionale, 1 sola volta) Specifica l’unità da usare per l’intervallo.

<IntervalOffset>Float</IntervalOffset> (opzionale, default=0, 1 sola volta) Specifica l’offset per il primo stripline dal minimo

<IntervalOffsetType>String\*</IntervalOffsetType> (opzionale, default=false, 1 sola volta) Specifica l’unità da usare per l’IntervalOffset.

<StripWidth>Float</StripWidth> (opzionale, default=0, 1 sola volta) Specifica la larghezza degli strip lines

<StripWidthType>String\*</StripWidthType> (opzionale, 1 sola volta) Specifica l’unità da usare per l’intervallo.

</ChartStripLines>

<Arrows>String</Arrows> (opzionale, 1 sola volta) **None** (default): Specifies no arrows. **Triangle**: Specifies triangle arrows. **SharpTriangle**: Specifies sharp triangle arrows. **Lines**: Specifies lines only.

<Scalar>Boolean</Scalar> (opzionale, default=false, 1 sola volta) Specifica se i valori sull’asse che sono essere mostrati sono scalari. True se e solo se è presente un asse in ChartCategoryAxes AND ogni ChartSeries contiene un ChartDataPoint con il valore X definito.

<Minimum>Numeric or DateTime</Minimum> (opzionale, 1 sola volta) Specifica il valore minimo per l’asse

<Maximum>Numeric or DateTime</Maximum> (opzionale, 1 sola volta) Specifica il valore massimo per l’asse

<LogScale>Boolean</LogScale> (opzionale, default=false, 1 sola volta) specifica se un asse è logaritmico

<LogBase>Float>2</LogBase> (opzionale, default=10, 1 sola volta) la base logaritmica

<HideLabels>Boolean</HideLabels> (opzionale, default=false, 1 sola volta) specifica se le atichesse dell’asse sono nascoste

<Angle>-90≤Float≤90</Angle> (opzionale, default=0, 1 sola volta) Specifica a quale angolatura mostrare le labels dell’asse

<PreventFontShrink>Boolean</PreventFontShrink> (opzionale, default=false, 1 sola volta) Specifica se la grandezza del font di un label non deve essere ridotto per entrare nel grafico

<PreventFontGrow>Boolean</PreventFontGrow> (opzionale, default=false, 1 sola volta) Specifica se la grandezza del font di un label non deve essere aumentato per entrare nel grafico

<PreventLabelOffset>Boolean</PreventLabelOffset> (opzionale, default=false, 1 sola volta) specifies whether axis labels MUST be staggered to fit within a Chart

<PreventWordWrap>Boolean</PreventWordWrap> (opzionale, default=false, 1 sola volta) specifies whether axis labels MUST NOT be word-wrapped to fit within a Chart.

<AllowLabelRotation>String</AllowLabelRotation> (opzionale, 1 sola volta) **Rotate90** (default): Specifies that axis labels may rotate in 90-degree increments. **Rotate30**: Specifies that axis labels may rotate in 30-degree increments. **Rotate45**: Specifies that axis labels may rotate in 45-degree increments. **None**: Specifies that axis labels MUST NOT rotate.

<IncludeZero>Boolean</IncludeZero> (opzionale, default=true, 1 sola volta) Specifica se l’asse deve sempre includere il valore 0

<LabelsAutoFitDisabled>Boolean</LabelsAutoFitDisabled> (opzionale, default=false, 1 sola volta) Specifica se le etichette non devono automaticamente ridimensionarsi per entrare nel grafico

<MinFontSize>RdlSize</MinFontSize> (opzionale, default=6, 1 sola volta) Specifica la grandezza minima del font quando è abilitato il LabelsAutoFit

<MaxFontSize>RdlSize</MaxFontSize> (opzionale, default=10, 1 sola volta) Specifica la grandezza massima del font quando è abilitato il LabelsAutoFit

<OffsetLabels>Boolean</OffsetLabels> (opzionale, default=false, 1 sola volta) Specifica se le etichette dell’asse devono essere offset

<HideEndLabels>Boolean</HideEndLabels> (opzionale, default=false, 1 sola volta) Specifica se le labels sono nascoste alla fine dell’asse

<ChartAxisScaleBreak> (opzionale, 1 sola volta) specifies scale break behavior and style for a ChartAxis.

<Enabled>Boolean</Enabled> (opzionale, default=false, 1 sola volta) Specifica se lo ScaleBreak è abilitato

<BreakLineType>String</BreakLineType> (opzionale, 1 sola volta) **Ragged** (default): Specifies that the display is a ragged line. **Straight**: Specifies that the display is a straight line. **Wave**: Specifies that the display is a wavy line. **None**: Specifies that a line for the scale break is not displayed.

<CollapsibleSpaceThreshold>9<Integer<91</CollapsibleSpaceThreshold> (opzionale, 1 sola volta) Specifica la percentuale di spazio vuoto prima che si attivi lo scale break

<MaxNumberOfBreaks>0<Integer<6</MaxNumberOfBreaks> (opzionale, default=2, 1 sola volta) Specifica il numero massimo di Scale Break applicati

<Spacing>0≤Float≤10</Spacing> (opzionale, default=1.5, 1 sola volta) Specifica l’ammontare di spazio da lasciare per lo scale break in percentuale della grandezza del grafico

<IncludeZero>String</IncludeZero> (opzionale, 1 sola volta) **Auto** (default): Specifies whether scale breaks MUST be allowed to span zero based on the data plotted against the axis. **True**: Specifies that a scale break MUST NOT be allowed to span zero. **False**: Specifies that a scale break is allowed to span zero.

<[Style](#Style)>...</Style> (opzionale, 1 sola volta)

</ChartAxisScaleBreak>

<[CustomProperties](#CustomProperties)>...</CustomProperties> (opzionale, 1 sola volta)

**</ChartAxis>**

String\*

**Auto (**default**)**: Specifies that the interval unit MUST be determined based on the data plotted against the axis. **Number**: Specifies that the interval is numeric. **Years**, **Months**, **Weeks**, **Days**, **Hours**, **Minutes**, **Seconds**, **Milliseconds**

**<ChartGridLines> (opzionale, 1 sola volta) Specifica le gridlines sull’asse del grafico**

<Enabled>String</Enabled> (opzionale, 1 sola volta) **True**: Specifies that gridlines MUST appear. **False**: Specifies that gridlines MUST NOT appear. **Auto** (default): Specifies that this value is interpreted as true for ChartAxis.ChartMajorGridLines and as false for ChartAxis.ChartMinorGridLines.

<[Style](#Style)>...</Style> (opzionale, 1 sola volta)

<Interval>Float</Interval> (opzionale, default=0, 1 sola volta) Specifica l’intervallo tra linee di griglia

<IntervalType>String\*</IntervalType> (opzionale, 1 sola volta) Specifica l’unità da usare per l’intervallo.

<IntervalOffset>Float</IntervalOffset> (opzionale, default=0, 1 sola volta) Specifica l’offset per il primo gridline dal minimo

<IntervalOffsetType>String\*</IntervalOffsetType> (opzionale, default=false, 1 sola volta) Specifica l’unità da usare per l’IntervalOffset.

**</ChartGridLines>**

String\*

**Auto (**default**)**: Specifies that the interval unit MUST be determined based on the data plotted against the axis. **Number**: Specifies that the interval is numeric. **Years**, **Months**, **Weeks**, **Days**, **Hours**, **Minutes**, **Seconds**, **Milliseconds**

**<ChartTickMarks> (opzionale, 1 sola volta) Specifica i segni di spunta sull’asse del grafico**

<Enabled>String</Enabled> (opzionale, 1 sola volta) **True:** Specifies that tick marks MUST appear. **False**: Specifies that tick marks MUST NOT appear. **Auto (**default**):** Specifies that this value is interpreted as true for ChartAxis.ChartMajorTickMarks and false for ChartAxis.ChartMinorTickMarks.

<Type>String</Type> (opzionale, 1 sola volta) **Outside (**default**)**: Specifies that tick marks MUST appear outside the axis. **Inside**: Specifies that tick marks MUST appear inside the axis. **Cross**: Specifies that tick marks MUST appear across the axis. **None**: Specifies that tick marks MUST NOT appear.

<[Style](#Style)>...</Style> (opzionale, 1 sola volta)

<Length>Float</Length> (opzionale, default=1, 1 sola volta) Specifica la lunghezza del segno di spunta in percentuale con la grandezza del grafico

<Interval>Float</Interval> (opzionale, default=0, 1 sola volta) Specifica l’intervallo tra i segni di spunta

<IntervalType>String\*</IntervalType> (opzionale, 1 sola volta) Specifica l’unità da usare per l’intervallo.

<IntervalOffset>Float</IntervalOffset> (opzionale, default=0, 1 sola volta) Specifica l’offset per il primo gridline dal minimo

<IntervalOffsetType>String\*</IntervalOffsetType> (opzionale, default=false, 1 sola volta) Specifica l’unità da usare per l’IntervalOffset.

**</ChartTickMarks>**

String\*

**Auto (**default**)**: Specifies that the interval unit MUST be determined based on the data plotted against the axis. **Number**: Specifies that the interval is numeric. **Years**, **Months**, **Weeks**, **Days**, **Hours**, **Minutes**, **Seconds**, **Milliseconds**

**<ChartElementPosition> (opzionale, 1 sola volta) Specifica la posizione dove disegnare gli elementi del grafico**

<Top>0≤Float≤100</Top> (opzionale, default=0, 1 sola volta) Specifica la distanza dell’item del grafico dalla parte superiore del contenitore come percentuale

<Left>0≤Float≤100</Left> (opzionale, default=0, 1 sola volta) Specifica la distanza dell’item del grafico dalla parte sinistra del contenitore come percentuale

<Height>0≤Float≤100</Height> (opzionale, default=0, 1 sola volta) Specifica l’altezza dell’item del grafico come percentuale dell’altezza del contenitore

<Width>0≤Float≤100</Width> (opzionale, default=0, 1 sola volta) Specifica la larghezza dell’item del grafico come percentuale della larghezza del contenitore

**</ChartElementPosition>**

**<ChartLegends> (opzionale, 1 sola volta) Specifica una collezione di legende per il grafico**

<ChartLegend Name="String"> (Identificatore)

<Hidden>Boolean</Hidden> (opzionale, default=false, 1 sola volta) Specifica se nascondere la legenda

             <[Style](#Style)>...</Style> (opzionale, 1 sola volta)

<Position>String\*</Position> (opzionale, 1 sola volta) Posizione della legenda

<Layout>String</Layout> (opzionale, 1 sola volta) **AutoTable** (default): Specifies that labels MUST be arranged automatically to fit. **Column**: Specifies that labels MUST be arranged in a column. **Row**: Specifies that labels MUST be arranged in a row. **WideTable**: Specifies that labels MUST be arranged in a wide table. **TallTable**: Specifies that labels MUST be arranged in a tall table.

<DockToChartArea>Boolean</DockToChartArea> (opzionale, false, 1 sola volta) Specifica il nome dell’area del grafico dove mostrare la legenda

<DockOutsideChartArea>Boolean</DockOutsideChartArea> (opzionale, default=false, 1 sola volta) Specifica se la legenda deve essere tagliata oltre l’area del grafico

<[ChartElementPosition](#ChartElementPosition)>...</ChartElementPosition> (opzionale, 1 sola volta) Specifica una posizione custom per la legenda

<[ChartLegendTitle](#ChartLegendTitle)>...</ChartLegendTitle> (opzionale, 1 sola volta) Specifica il titolo della legenda

<AutoFitTextDisabled>Boolean</AutoFitTextDisabled> (opzionale, default=false, 1 sola volta) Specifica se il testo può auto ridimensionarsi per entrare nell’area della legenda

<MinFontSize>RdlSize>5pt</MinFontSize> (USARE SOLO SE [AutoFitTextDisabled=false], opzionale, default=7pt, 1 sola volta) Specifica il valore minimo della grandezza usato dall’algoritmo di auto ridimensionamento

<HeaderSeparator>String\*\*</HeaderSeparator> (opzionale, default=None, 1 sola volta)

<HeaderSeparatorColor>String</HeaderSeparatorColor> (opzionale, default=empty, 1 sola volta) Specifica il colore del separatore

<ColumnSeparator>String\*\*</ColumnSeparator> (opzionale, default=None, 1 sola volta)

<ColumnSeparatorColor>String</ColumnSeparatorColor> (opzionale, default=empty, 1 sola volta) Specifica il colore del separatore

<ColumnSpacing>0≤Integer≤100</ColumnSpacing> (opzionale, default=50, 1 sola volta) Specifica lo spazio tra le colonne della legenda come percentuale della grandezza del font

<InterlacedRows>Boolean</InterlacedRows> (opzionale, default=false, 1 sola volta) Specifica se le righe della legenda usano colori interlacciati

<InterlacedRowsColor>String</InterlacedRowsColor> (opzionale, default=empty, 1 sola volta) Specifica il colore delle righe interlacciate

<EquallySpacedItems>Boolean</EquallySpacedItems> (opzionale, default=false, 1 sola volta) Specifica se gli ogetti della legenda hanno uguale spaziatura tra loro

<Reversed>String</Reversed> (opzionale, 1 sola volta) **Auto** (default): Specifies that the direction MUST be automatically detected based on the series types (specified by ChartSeries.Type and ChartSeries.Subtype). **True**: Specifies that the order of items in the legend MUST be reversed. **False**: Specifies that the order of items in the legend MUST NOT be reversed.

<MaxAutoSize>0≤Integer≤100</MaxAutoSize> (opzionale, default=50, 1 sola volta) Specifica la grandezza massima della legenda come percentuale della grandezza del grafico

<TextWrapThreshold>Integer>0</TextWrapThreshold> (opzionale, default=25, 1 sola volta) Specifies the number of characters after which to wrap text in a ChartLegend.

</ChartLegend>

**</ChartLegends>**

String\*

* **RightTop** (default: Positions the legend at the upper corner of the right side of the chart.
* **TopLeft**: Positions the legend at the left corner of the upper side of the chart.
* **TopCenter**: Positions the legend at the middle of the upper side of the chart.
* **TopRight**: Positions the legend at the right corner of the upper side of the chart.
* **LeftTop**: Positions the legend at the upper corner of the left side of the chart.
* **LeftCenter**: Positions the legend at the middle of the left side of the chart.
* **LeftBottom**: Positions the legend at the bottom corner of the left side of the chart.
* **RightCenter**: Positions the legend at the middle of the right side of the chart.
* **RightBottom**: Positions the legend at the bottom corner of the right side of the chart.
* **BottomRight**: Positions the legend at the right corner of the bottom side of the chart.
* **BottomCenter**: Positions the legend at the middle of the bottom side of the chart.
* **BottomLeft**: Positions the legend at the left corner of the bottom side of the chart.

String\*\*

**None**: Specifies no separator.

**Line**: Specifies to separate with a line.

**ThickLine**: Specifies to separate with a thick line.

**DoubleLine**: Specifies to separate with a double line.

**DashLine**: Specifies to separate with a dashed line.

**Dotline**: Specifies to separate with a dotted line.

**GradientLine**: Specifies to separate with a gradient line.

**ThickGradientLine**: Specifies to separate with a thick gradient line.

**<ChartLegendTitle> (opzionale, 1 sola volta) Specifica il titolo della legenda**

               <Caption>String</Caption> (Obbligatorio, 1 sola volta) Specifica la discalia del titolo

               <TitleSeparator>String\*</TitleSeparator> (opzionale, default=None, 1 sola volta)

               <[Style](#Style)>...</Style> (opzionale, 1 sola volta)

**</ChartLegendTitle>**

String\*

**None**: Specifies no separator.

**Line**: Specifies to separate with a line.

**ThickLine**: Specifies to separate with a thick line.

**DoubleLine**: Specifies to separate with a double line.

**DashLine**: Specifies to separate with a dashed line.

**Dotline**: Specifies to separate with a dotted line.

**GradientLine**: Specifies to separate with a gradient line.

**ThickGradientLine**: Specifies to separate with a thick gradient line.

Codice esempio

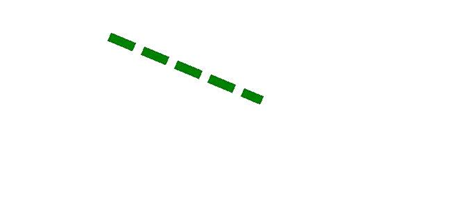
Consiglio di vedere quelli riportati in questi link:

<https://msdn.microsoft.com/en-us/library/ee224306(v=sql.105).aspx>

<https://msdn.microsoft.com/en-us/library/ee239893(v=sql.105).aspx>

### Line

Elememento opzionale che può essere inserito più di una volta. Disegna una linea.



 <Line Name="String">

  <[Style](#Style)>...</Style> (opzionale, 1 sola volta)

<Top>RdlSize</Top> (opzionale, 1 sola volta)

  <Left>RdlSize</Left> (opzionale, 1 sola volta)

<Height>RdlSize</Height> (opzionale, 1 sola volta)

  <Width>RdlSize</Width> (opzionale, 1 sola volta)

  <ZIndex>Integer</ZIndex> (opzionale, 1 sola volta)

  <[Visilibty](#Visibility)>...</Visibility> (opzionale, 1 sola volta)

  <DocumentMapLabel>String</DocumentMapLabel> (opzionale, 1 sola volta)

  <Bookmark>String</Bookmark> (opzionale, 1 sola volta)

  <[CustomProperties](#CustomProperties)>...</CustomProperties> (opzionale, 1 sola volta)

 </Line>

Codice esempio

 <Line Name="Line1">

  <Top>1.25917in</Top>

  <Left>4in</Left>

  <Height>-0.95833in</Height>

  <Width>-2.29167in</Width>

  <Style>

  <Border>

  <Color>Green</Color>

  <Style>Dashed</Style>

  <Width>10pt</Width>

  </Border>

  </Style>

 </Line>

Struttura generale Report in XML:

<?xml version="1.0" encoding="utf-8"?>

<Report xmlns="http://schemas.microsoft.com/sqlserver/reporting/2010/01/reportdefinition"> Indica la versione del report (attualmente questa è la più aggiornata che sono riuscito a far funzionare, con la 2016/01 non si apre)

<DataSources>

I DataSources del report

<DataSource Name="Elements">

<ConnectionProperties>

<DataProvider>ListOfT</DataProvider>

<ConnectString />

</ConnectionProperties>

</DataSource>

</DataSources>

<DataSets>

Collection of dataset

<DataSet Name="DataSet1">

<Query>

<DataSourceName>Elements</DataSourceName>

<CommandText />

</Query>

<Fields>

Elenco dei campi del dataset

<Field Name="Id1">

<DataField>Id1</DataField>

</Field>

<Field Name="Id2">

<DataField>Id2</DataField>

</Field>

<Field Name="Name">

<DataField>Name</DataField>

</Field>

<Field Name="Tipo">

<DataField>Tipo</DataField>

</Field>

</Fields>

</DataSet>

</DataSets>

Qui finisce la parte relativa ai data source e dataset

public class DataSources :IElement {

public const string DataSourceName = "Elements";

public XElement Element {

get {

return Build();

}

}

private static XElement Build() {

return new XElement(

typeof(DataSources).GetShortName(),

new XElement(

"DataSource",

new XAttribute("Name", DataSourceName),

new XElement(

"ConnectionProperties",

new XElement("DataProvider", "ListOfT"),

new XElement("ConnectString"))));

}

}

Inizio scrivendo i tag generali del datasource

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

public class DataSets :CollectionOf<DataSet>, IElement {

protected sealed override string GetRdlName() {

return typeof(DataSets).GetShortName();

}

}

Datasets come collezione di DataSet

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

public class DataSet :IElement {

private static int index;

private readonly List<string> fields;

private readonly string rdlName;

public DataSet() {

this.rdlName = typeof(DataSet).GetShortName();

this.Name = this.rdlName + ++index;

this.fields = new List<string>();

}

public string Name { get; private set; }

public XElement Element {

get {

return this.Build();

}

}

public void AddField(string field) {

this.fields.Add(field);

}

private XElement Build() {

var f = new XElement("Fields");

var dataSet = new XElement(

this.rdlName,

new XAttribute("Name", this.Name),

new XElement(

"Query",

new XElement("DataSourceName", DataSources.DataSourceName),

new XElement("CommandText")),

f);

foreach (var field in this.fields) {

f.Add(new XElement(

"Field",

new XAttribute("Name", field),

new XElement("DataField", field)));

}

return dataSet;

}

}

Index per indicare DataSet1,DataSet2, altrimenti va in errore.

Nel costruttore compongo il nome del Dataset e ci aggiungo una lista di stringhe (i campi del Dataset). Nel Build converto tutto in XML.

Nel Build() this.rdlName è necessario altrimenti si mangia il tag <Dataset>..</Dataset>

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

<ReportSections>

<ReportSection>

<Body>

<ReportItems>

Gli item da inserire nel report quali: Texbox, Linea, Tabella, Matrice, Rettangolo, Elenco, Immagine, SottoReport, Grafico, Misuratore, Mappa, Barra dei Dati, Grafico sparkline, Indicatore

<Tablix Name="Tablix1">

Item di tipo Tablix (Matrice)

<TablixBody>

<TablixColumns>

Collection of Column

<TablixColumn>

Il numero di TablixColumn deve essere = al numero dei TablixMember sotto la TablixColumnHyerarchy

<Width>1in</Width>

</TablixColumn>

</TablixColumns>

<TablixRows>

Collection of Row

<TablixRow>

Il numero di TablixRow deve essere = al numero dei TablixMember sotto la TablixRowHyerarchy

<Height>0.25in</Height>

<TablixCells>

Collection of Cell

<TablixCell>

Una riga con N celle necessita di avere una tablix con N colonne!

<CellContents>

<Textbox Name="Id1">

<CanGrow>true</CanGrow>

<KeepTogether>true</KeepTogether>

<Paragraphs>

Collection of paragraph

<Paragraph>

<TextRuns>

Collection of TextRuns

<TextRun>

Presenti attributi quale Value, FontFamily, FontSize, FontWeight, Color, MarkupType, Format

<Value>=Fields!Id1.Value</Value>

Può essere anche una stringa o una espressione.

Fields!X.Value è lo standard per i valori presenti nel DatasetN associato alla Tablix

<Style />

</TextRun>

</TextRuns>

<Style />

</Paragraph>

</Paragraphs>

<rd:DefaultName>Id1</rd:DefaultName>

<Style>

<Border>

<Color>LightGrey</Color>

<Style>Solid</Style>

</Border>

<PaddingLeft>2pt</PaddingLeft>

<PaddingRight>2pt</PaddingRight>

<PaddingTop>2pt</PaddingTop>

<PaddingBottom>2pt</PaddingBottom>

</Style>

</Textbox>

</CellContents>

</TablixCell>

</TablixCells>

</TablixRow>

</TablixRows>

</TablixBody>

<TablixColumnHierarchy>

Gerarchia delle colonne (group,sorting,value)

<TablixMembers>

Collection of member

I TablixMember devo essere di numero = al numero delle colonne della tablix altrimenti può andare in errore!!

<TablixMember />

</TablixMembers>

</TablixColumnHierarchy>

<TablixRowHierarchy>

Gerarchia delle righe (group,sorting,value)

<TablixMembers>

Collection of member

I TablixMember devo essere di numero = al numero delle righe della tablix altrimenti può andare in errore!!

<TablixMember />

</TablixMembers>

</TablixRowHierarchy>

<DataSetName>DataSet1</DataSetName>

Dataset usato dalla tablix

<Top>0.30083in</Top>

<Left>0.31125in</Left>

<Height>0.25in</Height>

<Width>1in</Width>

<Style>

<Border>

<Style>None</Style>

</Border>

</Style>

</Tablix>

</ReportItems>

Qui finisce la parte relativa ai report item

public class ReportItems :CollectionOf<ReportItem>, IElement {

protected sealed override string GetRdlName() {

return typeof(ReportItems).GetShortName();

}

}

Collezione di Reportitem

public abstract class ReportItem :IElement {

private Inch top = new Inch(0);

public Inch Top {

get {

return this.top;

}

set {

this.top = value;

}

}

public Inch Left { get; set; }

public Inch Height { get; set; }

public Inch Width { get; set; }

public Inch NextTop {

get {

return this.Top + this.Height;

}

}

public XElement Element {

get {

return this.Build();

}

}

protected abstract XElement Build();

protected void ConfigureTop(XElement item) {

if (this.Top.ToString() != new Inch(0).ToString()) {

item.Add(new XElement("Top", this.Top));

}

}

protected void ConfigureLeft(XElement item) {

if (this.Left != null) {

item.Add(new XElement("Left", this.Left));

}

}

protected void ConfigureHeight(XElement item) {

if (this.Height != null) {

item.Add(new XElement("Height", this.Height));

}

}

protected void ConfigureWidth(XElement item) {

if (this.Width != null) {

item.Add(new XElement("Width", this.Width));

}

}

}

public class Tablix :ReportItem {

private static int index;

private readonly string name;

private readonly TablixCorner tablixCorner;

private readonly TablixColumnHierarchy tablixColumnHierarchy;

private readonly TablixRowHierarchy tablixRowHierarchy;

private readonly TablixBody tablixBody;

private readonly string dataSetName;

private readonly string rdlName;

public Tablix(

TablixColumnHierarchy tablixColumnHierarchy,

TablixRowHierarchy tablixRowHierarchy,

TablixBody tablixBody,

string dataSetName) {

this.rdlName = typeof(Tablix).GetShortName();

this.name = this.rdlName + ++index;

this.tablixColumnHierarchy = tablixColumnHierarchy;

this.tablixRowHierarchy = tablixRowHierarchy;

this.tablixBody = tablixBody;

this.dataSetName = dataSetName;

}

public Tablix(

TablixCorner tablixCorner,

TablixColumnHierarchy tablixColumnHierarchy,

TablixRowHierarchy tablixRowHierarchy,

TablixBody tablixBody,

string dataSetName)

: this(tablixColumnHierarchy, tablixRowHierarchy, tablixBody, dataSetName) {

this.tablixCorner = tablixCorner;

}

public Style Style { get; set; }

protected override XElement Build() {

this.Height = this.tablixBody.Height + this.tablixColumnHierarchy.Height;

this.Width = this.tablixBody.Width + this.tablixRowHierarchy.Width;

var result = new XElement(

this.rdlName,

new XAttribute("Name", this.name),

this.tablixColumnHierarchy.Element,

this.tablixRowHierarchy.Element,

this.tablixBody.Element,

new XElement("DataSetName", this.dataSetName));

this.ConfigureTablixCorner(result);

this.ConfigureTop(result);

this.ConfigureLeft(result);

this.ConfigureHeight(result);

this.ConfigureWidth(result);

this.ConfigureStyle(result);

return result;

}

private void ConfigureTablixCorner(XElement tablix) {

if (this.tablixCorner != null) {

tablix.Add(this.tablixCorner.Element);

}

}

private void ConfigureStyle(XElement tablix) {

if (this.Style != null) {

tablix.Add(this.Style.Element);

}

}

}

Nel costruttore richiesto i campi (che poi saranno convertiti in TAG XML) obbligatori per comporre la matrice.

Nel Build converto tutto in XML. tablixColumnHierarchy, tablixRowHierarchy, tablixBody li posso mettere .Element in quanto sono già stati elaborati e convertiti in precedenza

public class TablixBody {

private readonly TablixColumns tablixColumns;

private readonly TablixRows tablixRows;

public TablixBody(TablixColumns tablixColumns, TablixRows tablixRows) {

this.tablixColumns = tablixColumns;

this.tablixRows = tablixRows;

}

public Inch Height {

get {

return this.tablixRows.Height;

}

}

public Inch Width {

get {

return this.tablixColumns.Width;

}

}

public XElement Element {

get {

return this.Build();

}

}

private XElement Build() {

return new XElement(typeof(TablixBody).GetShortName(), this.tablixColumns.Element, this.tablixRows.Element);

}

}

Corpo della tablix. Nel costruttore richiesto le collezioni di colonne e le collezioni di righe.

Nel Build converto tutto in XML. tablixColumns, tablixRows li posso mettere .Element in quanto sono già stati elaborati e convertiti in precedenza

public class TablixColumns :CollectionOf<TablixColumn>, IElement {

public TablixColumns() {

}

public TablixColumns(TablixColumn tablixColumn)

: base(tablixColumn) {

}

public Inch Width {

get {

var result = new Inch(0);

return this.Collection.Aggregate(result, (current, tablixColum) => current + tablixColum.Width);

}

}

}

Collezione di Colonne. Richiedo nel costruttore la colonna in quanto almeno 1 colonna è obbligatoria.

Aggiunto anche costruttore senza richiesta di colonna. Necessario però fare TablixColumns.Add(…);

public class TablixColumn :ParentOf<Width> {

public TablixColumn(Width item)

: base(item) {

}

public Inch Width {

get {

return this.Item.Value;

}

}

protected sealed override string GetRdlName() {

return typeof(TablixColumn).GetShortName();

}

}

public class TablixRows :CollectionOf<TablixRow>, IElement {

public TablixRows() {

}

public TablixRows(TablixRow tablixRow) {

this.Add(tablixRow);

}

public Inch Height {

get {

var result = new Inch(0);

return this.Collection.Aggregate(result, (current, tablixRow) => current + tablixRow.Height);

}

}

protected sealed override string GetRdlName() {

return typeof(TablixRows).GetShortName();

}

}

Collezione di Righe. Richiedo nel costruttore la riga in quanto almeno 1 riga è obbligatoria.

Aggiunto anche costruttore senza richiesta di riga. Necessario però fare TablixRows.Add(…);

public class TablixRow :IElement {

private readonly TablixCells tablixCells;

public TablixRow(Inch height, TablixCells tablixCells) {

this.Height = height;

this.tablixCells = tablixCells;

}

public Inch Height { get; private set; }

public XElement Element {

get {

return this.Build();

}

}

private XElement Build() {

return new XElement(

typeof(TablixRow).GetShortName(),

new XElement("Height", this.Height),

this.tablixCells.Element);

}

}

La tablix row può contenere 1->N celle, per questo nel costruttore è obbligatorio inserire la collezione di celle

public class TablixCells :CollectionOf<TablixCell>, IElement {

public TablixCells() {

}

public TablixCells(TablixCell tablixCell)

: base(tablixCell) {

}

protected sealed override string GetRdlName() {

return typeof(TablixCells).GetShortName();

}

}

Collezione di celle. Richiedo nel costruttore la cella in quanto almeno 1 cella è obbligatoria.

Aggiunto anche costruttore senza richiesta di cella. Necessario però fare TablixCells.Add(…);

public class TablixCell :ParentOf<CellContents> {

public TablixCell(CellContents item)

: base(item) {

}

protected sealed override string GetRdlName() {

return typeof(TablixCell).GetShortName();

}

}

Cella di Tablix. Nel costruttore passo il contenuto della cella (può essere anche una matrice stessa!)

public class CellContents :ParentOf<ReportItem> {

public CellContents(ReportItem reportItem)

: base(reportItem) {

}

protected sealed override string GetRdlName() {

return typeof(CellContents).GetShortName();

}

}

public class Textbox :ReportItem {

private static int index;

private readonly string name;

private readonly Paragraphs paragraphs = new Paragraphs();

private readonly string rdlName;

public Textbox(Paragraph paragraph) {

this.rdlName = typeof(Textbox).GetShortName();

this.name = this.rdlName + ++index;

this.AddParagraph(paragraph);

}

public TextboxStyle TextboxStyle { get; set; }

public Visibility Visibility { get; set; }

public void AddParagraph(Paragraph paragraph) {

this.paragraphs.Add(paragraph);

}

protected override XElement Build() {

var result = new XElement(

this.rdlName,

new XAttribute("Name", this.name),

new XElement("CanGrow", true),

new XElement("KeepTogether", true),

this.paragraphs.Element);

this.ConfigureVisibility(result);

this.ConfigureTextboxStyle(result);

this.ConfigureTop(result);

this.ConfigureLeft(result);

this.ConfigureHeight(result);

this.ConfigureWidth(result);

return result;

}

private void ConfigureTextboxStyle(XElement textbox) {

if (this.TextboxStyle != null) {

textbox.Add(this.TextboxStyle.Element);

}

}

private void ConfigureVisibility(XElement textbox) {

if (this.Visibility != null) {

textbox.Add(this.Visibility.Element);

}

}

}

La classe del texbox funziona in quandi linee come quella della matrice.

Richiedo il paragraph nel costruttore il quale verrà aggiunto alla collection Paragraphs. Nel Build() faccio this.paragraphs.Element in quanto ho paragraphs già elaborato e convertito

public class Paragraphs :CollectionOf<Paragraph>, IElement {

public Paragraphs() {

}

public Paragraphs(Paragraph paragraph)

: base(paragraph) {

}

protected sealed override string GetRdlName() {

return typeof(Paragraphs).GetShortName();

}

}

public class Paragraph :ParentOf<TextRuns> {

public Paragraph(TextRuns item)

: base(item) {

}

public TextAlign TextAlign { get; set; }

protected sealed override string GetRdlName() {

return typeof(Paragraph).GetShortName();

}

protected override XElement Build() {

var result = base.Build();

result.Add(this.BuildStyle());

return result;

}

private XElement BuildStyle() {

var result = new XElement("Style");

if (this.TextAlign != TextAlign.Default) {

result.Add(new XElement("TextAlign", this.TextAlign));

}

return result.HasElements ? result : null;

}

}

public class TextRuns :CollectionOf<TextRun>, IElement {

public TextRuns(TextRun textRun)

: base(textRun) {

}

protected sealed override string GetRdlName() {

return typeof(TextRuns).GetShortName();

}

}

public class TextRun :IElement {

public string Value { get; set; }

public string FontFamily { get; set; }

public Point FontSize { get; set; }

public FontWeight FontWeight { get; set; }

public string Color { get; set; }

public MarkupType MarkupType { get; set; }

public string Format { get; set; }

public XElement Element {

get {

return this.Build();

}

}

private XElement Build() {

var result = new XElement(typeof(TextRun).GetShortName(), new XElement("Value", this.Value));

this.ConfigureMarkupType(result);

this.ConfigureStyle(result);

return result;

}

private void ConfigureStyle(XElement textRun) {

var style = new XElement("Style");

if (this.FontFamily != null) {

style.Add(new XElement("FontFamily", this.FontFamily));

}

if (this.FontSize != null) {

style.Add(new XElement("FontSize", this.FontSize));

}

if (this.FontWeight != FontWeight.Default) {

style.Add(new XElement("FontWeight", this.FontWeight));

}

if (this.Color != null) {

style.Add(new XElement("Color", this.Color));

}

if (this.Format != null) {

style.Add(new XElement("Format", this.Format));

}

if (style.HasElements) {

textRun.Add(style);

}

}

private void ConfigureMarkupType(XElement textRun) {

if (this.MarkupType != MarkupType.None) {

textRun.Add(new XElement("MarkupType", this.MarkupType));

}

}

}

Nel texrun si può settare Style, FontFamily, FontSize, FontWeight, Color, Format di un qualsiasi textbox.

public class Style :IElement {

public Border Border { get; set; }

public LeftBorder LeftBorder { get; set; }

public RightBorder RightBorder { get; set; }

public TopBorder TopBorder { get; set; }

public BottomBorder BottomBorder { get; set; }

public string BackgroundColor { get; set; }

public BackgroundImage BackgroundImage { get; set; }

public XElement Element {

get {

return this.Build();

}

}

protected virtual XElement Build() {

var result = new XElement(typeof(Style).GetShortName());

this.ConfigureBorder(result);

this.ConfigureTopBorder(result);

this.ConfigureBottomBorder(result);

this.ConfigureLeftBorder(result);

this.ConfigureRightBorder(result);

this.ConfigureBackgroundColor(result);

this.ConfigureBackgroundImage(result);

return result;

}

private void ConfigureBorder(XElement style) {

if (this.Border != null) {

style.Add(this.Border.Element);

}

}

private void ConfigureTopBorder(XElement style) {

if (this.TopBorder != null) {

style.Add(this.TopBorder.Element);

}

}

private void ConfigureBottomBorder(XElement style) {

if (this.BottomBorder != null) {

style.Add(this.BottomBorder.Element);

}

}

private void ConfigureLeftBorder(XElement style) {

if (this.LeftBorder != null) {

style.Add(this.LeftBorder.Element);

}

}

private void ConfigureRightBorder(XElement style) {

if (this.RightBorder != null) {

style.Add(this.RightBorder.Element);

}

}

private void ConfigureBackgroundColor(XElement style) {

if (this.BackgroundColor != null) {

style.Add(new XElement("BackgroundColor", this.BackgroundColor));

}

}

private void ConfigureBackgroundImage(XElement style) {

if (this.BackgroundImage != null) {

style.Add(this.BackgroundImage.Element);

}

}

}

public class Border :IElement {

public object Style { get; set; }

public string Color { get; set; }

public Point Width { get; set; }

public XElement Element {

get {

return this.Build();

}

}

protected virtual string BorderName {

get {

return "Border";

}

}

private XElement Build() {

if (this.Style == null) {

return null;

}

var result = new XElement(this.BorderName);

result.Add(new XElement("Style", this.Style));

if (this.Color != null) {

result.Add(new XElement("Color", this.Color));

}

if (this.Width != null) {

result.Add(new XElement("Width", this.Width));

}

return result;

}

}

public class TablixColumnHierarchy {

private readonly TablixMembers tablixMembers;

public TablixColumnHierarchy(TablixMembers tablixMembers) {

this.tablixMembers = tablixMembers;

}

public Inch Height {

get {

return this.tablixMembers.Size;

}

}

public XElement Element {

get {

return this.Build();

}

}

private XElement Build() {

return new XElement(typeof(TablixColumnHierarchy).GetShortName(), this.tablixMembers.Element);

}

Gerarchia delle colonne. Nel costruttore cheido i tablixMembers.

I TablixMember devo essere di numero = al numero delle colonne della tablix altrimenti può andare in errore!!

public class TablixMembers :CollectionOf<TablixMember>, IElement {

public TablixMembers() {

}

public TablixMembers(TablixMember tablixMember)

: base(tablixMember) {

}

public Inch Size {

get {

var result = new Inch(0);

return this.Collection.Aggregate(result, (current, tablixMember) => current + tablixMember.Size);

}

}

protected sealed override string GetRdlName() {

return typeof(TablixMembers).GetShortName();

}

}

Collezione di TablixMembers. Richiedo nel costruttore il TablixMember in quanto almeno 1 TablixMember è obbligatorio.

Aggiunto anche costruttore senza richiesta di TablixMember. Necessario però fare TablixMembers.Add(…);

public class TablixMember :IElement {

private readonly Group group;

private readonly SortExpressions sortExpressions;

private readonly TablixHeader tablixHeader;

public TablixMember() {

}

public TablixMember(Group group, SortExpressions sortExpressions, TablixHeader tablixHeader) {

this.group = group;

this.sortExpressions = sortExpressions;

this.tablixHeader = tablixHeader;

}

public Inch Size {

get {

return this.tablixHeader == null ? new Inch(0) : this.tablixHeader.Size;

}

}

public XElement Element {

get {

return this.Build();

}

}

private XElement Build() {

var result = new XElement(typeof(TablixMember).GetShortName());

this.ConfigureGroup(result);

this.ConfigureSortExpressions(result);

this.ConfigureTablixHeader(result);

return result;

}

private void ConfigureGroup(XElement tablixMember) {

if (this.group != null) {

tablixMember.Add(this.group.Element);

}

}

private void ConfigureSortExpressions(XElement tablixMember) {

if (this.sortExpressions != null) {

tablixMember.Add(this.sortExpressions.Element);

}

}

private void ConfigureTablixHeader(XElement tablixMember) {

if (this.tablixHeader != null) {

tablixMember.Add(this.tablixHeader.Element);

}

}

}

I TablixMember devo essere di numero = al numero delle colonne della tablix altrimenti può andare in errore!!

Per questo è possibile fare TablixMember “vuoti” new TablixMember(); in quanto non sempre è necessario inserirne uno configurato con raggruppamento, ordinamento e valore da mostrare.

public class TablixRowHierarchy :ParentOf<TablixMembers> {

public TablixRowHierarchy(TablixMembers item)

: base(item) {

}

public Inch Width {

get {

return this.Item.Size;

}

}

protected sealed override string GetRdlName() {

return typeof(TablixRowHierarchy).GetShortName();

} }

Come per le colonne

<rd:ReportUnitType>Inch</rd:ReportUnitType>

<rd:ReportID>0e470cab-117a-4b72-b771-c94c9aa83dc2</rd:ReportID>

</Report>

**ALTRI TAG:**

**Header e Footer**

<Page>

<PageHeader>

<Height>0.5in</Height>

<PrintOnFirstPage>true</PrintOnFirstPage>

<PrintOnLastPage>true</PrintOnLastPage>

<ReportItems>

Può contenere Items quali textbox, rectangle, image, linea

<Textbox Name="Textbox8">

<CanGrow>true</CanGrow>

<KeepTogether>true</KeepTogether>

<Paragraphs>

<Paragraph>

<TextRuns>

<TextRun>

<Value>Prova Header</Value>

</TextRun>

</TextRuns>

</Paragraph>

</Paragraphs>

<Style>

<PaddingTop>2pt</PaddingTop>

<PaddingBottom>2pt</PaddingBottom>

<PaddingLeft>2pt</PaddingLeft>

<PaddingRight>2pt</PaddingRight>

</Style>

</Textbox>

</ReportItems>

</PageHeader>

<PageFooter>

<Height>0.5in</Height>

<PrintOnFirstPage>true</PrintOnFirstPage>

<PrintOnLastPage>true</PrintOnLastPage>

<ReportItems>

Può contenere Items quali textbox, rectangle, image, linea

<Textbox Name="Textbox9">

<CanGrow>true</CanGrow>

<KeepTogether>true</KeepTogether>

<Paragraphs>

<Paragraph>

<TextRuns>

<TextRun>

<Value>Prova Footer</Value>

</TextRun>

</TextRuns>

</Paragraph>

</Paragraphs>

<Style>

<PaddingTop>2pt</PaddingTop>

<PaddingBottom>2pt</PaddingBottom>

<PaddingLeft>2pt</PaddingLeft>

<PaddingRight>2pt</PaddingRight>

</Style>

</Textbox>

</ReportItems>

</PageFooter>

<LeftMargin>1in</LeftMargin>

<RightMargin>1in</RightMargin>

<TopMargin>1in</TopMargin>

<BottomMargin>1in</BottomMargin>

</Page>

public class PageHeader:ReportItemsContainer {

public PageHeader(Inch Height, object PrintOnFirstPage, object PrintOnLastPage) {

this.height = Height;

this.printOnFirstPage = PrintOnFirstPage;

this.printOnLastPage = PrintOnLastPage;

Build();

}

public Inch height { get; private set; }

public object printOnFirstPage { get; private set; }

public object printOnLastPage { get; private set; }

protected override XElement Build() {

var result = new XElement(typeof(PageHeader).GetShortName());

this.Height(result);

this.PrintOnFirstPage(result);

this.PrintOnLastPage(result);

result.Add(this.ReportItems.Element);

return result;

}

private void Height(XElement PageHeader) {

if (this.height != null) {

PageHeader.Add(new XElement("Height", this.height));

}

}

private void PrintOnFirstPage(XElement PageHeader) {

if (this.printOnFirstPage != null) {

PageHeader.Add(new XElement("PrintOnFirstPage", this.printOnFirstPage));

}

}

private void PrintOnLastPage(XElement PageHeader) {

if (this.printOnLastPage != null) {

PageHeader.Add(new XElement("PrintOnLastPage", this.printOnLastPage));

}

}

}

public class PageFooter:ReportItemsContainer {

public PageFooter(Inch Height, object PrintOnFirstPage, object PrintOnLastPage) {

this.height = Height;

this.printOnFirstPage = PrintOnFirstPage;

this.printOnLastPage = PrintOnLastPage;

Build();

}

public Inch height { get; private set; }

public object printOnFirstPage { get; private set; }

public object printOnLastPage { get; private set; }

protected override XElement Build() {

var result = new XElement(typeof(PageFooter).GetShortName());

this.Height(result);

this.PrintOnFirstPage(result);

this.PrintOnLastPage(result);

result.Add(this.ReportItems.Element);

return result;

}

private void Height(XElement PageFooter) {

if (this.height != null) {

PageFooter.Add(new XElement("Height", this.height));

}

}

private void PrintOnFirstPage(XElement PageFooter) {

if (this.printOnFirstPage != null) {

PageFooter.Add(new XElement("PrintOnFirstPage", this.printOnFirstPage));

}

}

private void PrintOnLastPage(XElement PageFooter) {

if (this.printOnLastPage != null) {

PageFooter.Add(new XElement("PrintOnLastPage", this.printOnLastPage));

}

}

}

**Visibility e Drill Down (Toggleitem)**

Visibility e drill down fanno sempre parte delle proprietà che si possono associare al textbox. In <ToggleItem>Textbox1</ToggleItem> è necessario specificare il Textbox mediante il quale si espande o comprime il Texbox su cui vale la proprietà (-<Textbox Name="Textbox10">)

Su Hidden oltre a settare true o false è possibile inserire una espressione

-<Textbox Name="Textbox10">

<CanGrow>true</CanGrow>

<KeepTogether>true</KeepTogether>

+<Paragraphs>

-<Visibility>

<Hidden>true</Hidden>

<ToggleItem>Textbox1</ToggleItem>

</Visibility>

+<Style>

</Textbox>

public class Textbox :ReportItem {

public Visibility Visibility { get; set; }

protected override XElement Build() {

…

this.ConfigureVisibility(result);

…}

private void ConfigureActionInfo(XElement textbox) {

if (this.ActionInfo != null) {

textbox.Add(this.ActionInfo.Element);

}

}

public class Visibility {

/// <summary>

/// Funzione di drilldown. Hidden indica se la cella è visibile o meno, Toggleitem = textbox su cui vale la proprietà

/// </summary>

/// <param name="Hidden"></param>

/// <param name="Toggleitem"></param>

public Visibility(bool Hidden, string Toggleitem) {

this.hidden = Hidden;

this.toggleitem = Toggleitem;

Build();

}

/// <summary>

/// Funzione di drilldown. Hidden indica se la cella è visibile o meno, Toggleitem = textbox su cui vale la proprietà

/// </summary>

/// <param name="Hidden"></param>

/// <param name="Toggleitem"></param>

public Visibility(string Hidden, string Toggleitem) {

this.hidden = Hidden;

this.toggleitem = Toggleitem;

Build();

}

public XElement Element {

get {

return this.Build();

}

}

public object hidden { get; private set; }

public object toggleitem { get; private set; }

protected virtual XElement Build() {

var result = new XElement(typeof(Visibility).GetShortName());

this.Hidden(result);

this.Toggleitem(result);

return result;

}

private void Hidden(XElement Visibility) {

if (this.hidden != null) {

Visibility.Add(new XElement("Hidden", this.hidden));

}

}

private void Toggleitem(XElement Visibility) {

if (this.toggleitem != null) {

Visibility.Add(new XElement("ToggleItem", this.toggleitem));

}

}

}

**ActionInfo e DrillThrough**

L’uso dell’actionInfo con le actions lo ho trovato solo per il DrillThrough

-<Textbox Name="Textbox10">

<CanGrow>true</CanGrow>

<KeepTogether>true</KeepTogether>

+<Paragraphs>

<ActionInfo>

<Actions>

<Action>

<Drillthrough>

<ReportName>report.rdlc</ReportName>

</Drillthrough>

</Action>

</Actions>

</ActionInfo>

+<Style>

</Textbox>

public class ActionInfo {

private readonly Actions actions = new Actions();

private readonly string rdlName;

public ActionInfo(Action action) {

this.rdlName = typeof(ActionInfo).GetShortName();

this.AddAction(action);

}

public void AddAction(Action action) {

this.actions.Add(action);

}

public XElement Element {

get {

return this.Build();

}

}

protected virtual XElement Build() {

var result = new XElement(

this.rdlName,

new XElement(this.actions.Element));

return result;

}

}

public class Actions :CollectionOf<Action>, IElement {

public Actions() {

}

public Actions(Action action)

: base(action) {

}

protected sealed override string GetRdlName() {

return typeof(Actions).GetShortName();

}

}

public class Action :IElement {

public Drillthrough Drillthrough = new Drillthrough();

private readonly string rdlName;

public Action(string reportname) {

this.rdlName = typeof(Action).GetShortName();

this.Addreportname(reportname);

}

public void Addreportname(string reportname) {

this.Drillthrough.Add(reportname);

}

public XElement Element {

get {

return this.Build();

}

}

protected virtual XElement Build() {

var result = new XElement(

this.rdlName);

this.ConfigureDrillthrough(result);

return result;

}

private void ConfigureDrillthrough(XElement action) {

if (this.Drillthrough != null) {

action.Add(this.Drillthrough.Element);

}

}

}

public class Drillthrough {

public object reportname { get; private set; }

public XElement Element {

get {

return this.Build();

}

}

protected virtual XElement Build() {

var result = new XElement(typeof(Drillthrough).GetShortName());

this.Reportname(result);

return result;

}

internal void Add(string Reportname) {

this.reportname = Reportname;

this.Build();

}

private void Reportname(XElement Drillthrough) {

if (this.reportname != null) {

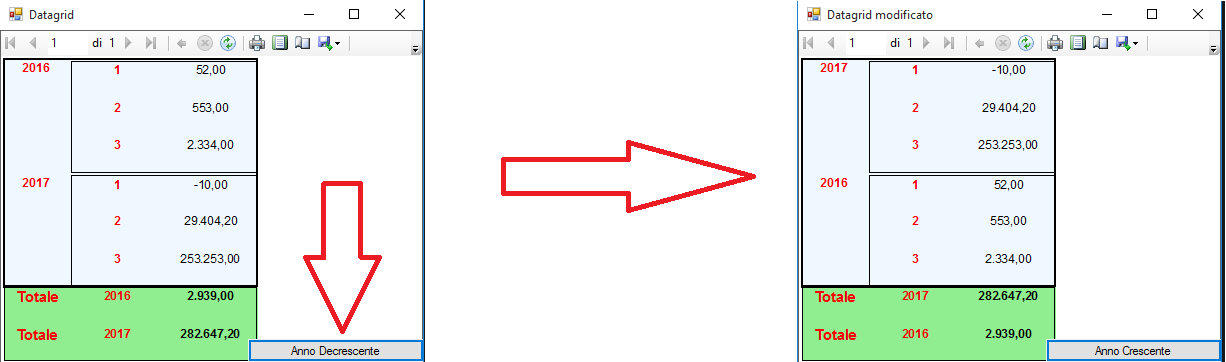
Drillthrough.Add(new XElement("ReportName", this.reportname));

}

}

}

**Aggiunta dei bottun** [**System.Windows.Forms**](https://msdn.microsoft.com/it-it/library/system.windows.forms.button(v=vs.110).aspx)



A quanto pare è possibile usare i button di [System.Windows.Forms](https://msdn.microsoft.com/it-it/library/system.windows.forms.button(v=vs.110).aspx) ed adattarli alle nostre esigenze come ad esempio mostrare un parametro in modo crescente o decrescente. Per fare ciò è necessario ovviamente rigenerare il report.

Per prima cosa ho aggiunto il button dalla finetra di progettazione.

Ho associato a questo 2 metodi sul click: CambiaAnno e Form1\_Load

//

// button1

//

this.button1.Anchor = ((System.Windows.Forms.AnchorStyles)((System.Windows.Forms.AnchorStyles.Bottom | System.Windows.Forms.AnchorStyles.Right)));

this.button1.Location = new System.Drawing.Point(248, 309);

this.button1.Name = "button1";

this.button1.Size = new System.Drawing.Size(176, 23);

this.button1.TabIndex = 0;

this.button1.Text = "Anno Decrescente";

this.button1.UseVisualStyleBackColor = true;

this.button1.Click += new System.EventHandler(this.CambiaAnno);

this.button1.Click += new System.EventHandler(this.Form1\_Load);

private bool decrescente { get; set; }

private void CambiaAnno(object sender, System.EventArgs e) {

this.Controls.Clear();

this.InitializeComponent();

this.Text = "Datagrid modificato";

if (decrescente == false) { button1.Text = "Anno Crescente"; this.decrescente = true;}

else { button1.Text = "Anno Decrescente"; this.decrescente = false; }

}

Questo metodo banalmente cambia la texbox e setta la variabile bool decrescente.

È necessario usare

this.Controls.Clear();

this.InitializeComponent();

altrimenti il form non mostra il report aggiornato, ma continua a mostrare la versione originale!!

private void Form1\_Load(object sender, System.EventArgs e) {

var rv = new ReportViewer { Dock = DockStyle.Fill };

this.Controls.Add(rv);

new DataGridReportGenerator(rv.LocalReport).Run(decrescente);

rv.ZoomMode = Microsoft.Reporting.WinForms.ZoomMode.FullPage;

rv.RefreshReport();

}

Viene istanziato il reportviewer e caricato con il report generato a run time.

Passo la variabile decrescente nel

new DataGridReportGenerator(rv.LocalReport).Run(decrescente);

ALTRE CLASSI

TablixRow

public static TablixRow CreateTablixRow(string Element, bool subtotal, string fontFamily, int font, FontWeight fontweight, string color, string format, TextAlign textAlign, double inch) {

var textRun = new TextRun {

Value = "=" + Expression.FieldsValue(Element, subtotal),

FontFamily = fontFamily,

FontSize = new Point(font),

FontWeight = fontweight,

Color = color,

Format = format

};

var paragraph = new Paragraph(new TextRuns(textRun)) { TextAlign = textAlign };

var textbox = new Textbox(paragraph) { TextboxStyle = new TextboxStyle() };

var tablixCells = new TablixCells(new TablixCell(new CellContents(textbox)));

return new TablixRow(new Inch(inch), tablixCells);

}

Crea una riga Tablix in una Tablix. Element = campo da visualizzare, subtotal = indica se è subtotale, font = dimensione del font, fontFamily = tipo font, fontweight = spessore, color = colore, format = formato, textAlign = allieamento testo, inch = distanza tra le righe

Per il parametro Element si può usare una qualsiasi stringa

Per il parametro fontFamily un si può usare System.Drawing.FontFamily.X.Name

Per il parametro font un si può usare int > 0

Per il parametro fontweight si può usare FontWeight.

Per il parametro color si può usare RdlColor.

Per il parametro format si può usare RdlColor.

Per il parametro TextAlign si può usare TextAlign.

Per il parametro inch si può usare un double > 0

public enum FontWeight {

Default,

public enum RdlColor {

Black,

public enum TextAlign {

Default,

CreateTablixRow utilizza la classe TextRun

La quale si occupa di trasformare i dati in XML tramite XElement di System.Xml.Linq;

Per il campo Value che si passa al TextRun si passa per la classe Expression la quale codifica i inseriti

public static string FieldsValue(string field,bool subtotal) {

string fieldsvalue;

if (subtotal) { fieldsvalue = "SUM(Fields!" + field + ".Value)"; }

else { fieldsvalue = "Fields!" + field + ".Value"; }

return fieldsvalue;

}

TablixStyle

Questa classe con il metodo CreateTablixStyle dà la possibilità di assegnare uno stile alla cella della matrice

public static Style CreateTablixStyle(string color, string style, int grandezza, string BackgroundColor) {

var border = new Border {

Color = color,

Style = style,

Width = new Point(grandezza),

};

return new Style {

Border = border,

BackgroundColor = BackgroundColor

};

}

Per il parametro color si può usare RdlColor.

Per il parametro style si può usare BorderStyle.

Per il parametro grandezza un int >= 0

Per il parametro BackgroundColor vanno bene passare le palette di .net (System.Drawing.Color.X.Name) come stringhe eg. "AliceBlue","PaleTurquoise"

public enum BorderStyle {

Default,

PER CREARE UN REPORT

Creare un form ed inserirci:

private void Form1\_Load(object sender, System.EventArgs e) {

var rv = new ReportViewer { Dock = DockStyle.Fill };

this.Controls.Add(rv);

new DataGridReportGenerator(rv.LocalReport).Run();

rv.RefreshReport();

}

Associato all’evento load del form

this.Load += new System.EventHandler(this.Form1\_Load);

Questo serve ad istanziare il ReportViewer per visualizzare il report e la classe DataGridReportGenerator che avvia tutto il progetto di costruzione

public class DataGridReportGenerator :ReportGenerator {

public DataGridReportGenerator(LocalReport localReport)

: base(localReport) {

}

public override void Run() {

var dataSetForMainTable = CreateDataSet();

var mainTable = MainTable.Create(dataSetForMainTable.Name);

var body = new Body();

body.AddReportItem(mainTable);

this.Report.AddReportSection(new ReportSection(body));

this.Report.AddDataSet(dataSetForMainTable);

this.DataSources.Add(new ReportDataSource(dataSetForMainTable.Name, MainTableDataSource.Create()));

base.Run();

}

private static DataSet CreateDataSet() {

var dataSet = new DataSet();

dataSet.AddField(ElementProperty.Id1);

dataSet.AddField(ElementProperty.Id2);

dataSet.AddField(ElementProperty.Name);

dataSet.AddField(ElementProperty.Tipo);

return dataSet;

}

}

NEL public override void Run() {

Si definisce il dataset, la maintable (la schermata principale che visualizzeremo), il body (al quale si passa la maintable), il datasource

IN private static DataSet CreateDataSet() {

Si aggiungono tutti i vari dataset (IN REALTA’ SONO DATASET CON 1 SOLA TABELLA, QUINDI COME FOSSERO DATATABLE)

Questa classe estende ReportGenerator che genera il report in formato XML (include alcune funzioni per il debug)

public class ReportGenerator {

protected readonly Report.Rdl.Report Report = new Report.Rdl.Report();

protected readonly ReportDataSourceCollection DataSources;

private readonly LocalReport localReport;

public ReportGenerator(LocalReport localReport) {

if (localReport == null) {

throw new ArgumentNullException("localReport");

}

this.localReport = localReport;

this.DataSources = localReport.DataSources;

}

public virtual void Run() {

////this.Report.Element.Save(Console.Out); // Utile per il debug se si vuole vedere il report il console.

////Console.WriteLine(Report.Element); // Utile per il debug se si vuole vedere il report il console.

this.Report.Element.Save("x.rdlc"); //Ti salva il report in …Creport\Creport\bin\Debug

this.LoadReportDefinition();

}

private void LoadReportDefinition() {

using (var stream = new MemoryStream()) {

this.Report.Element.Save(stream);

stream.Position = 0;

this.localReport.LoadReportDefinition(stream);

}

}

}

MainTable

È la schermata principale del report

public static ReportItem Create(string dataSetName) {

return new Tablix(

TablixColumnHierarchy.CreateTablixColumnHierarchy(),

TablixRowHierarchy.CreateTablixRowHierarchy(),

CreateTablixBody(),

dataSetName) {

Top = new Inch(0.025),

Left = new Inch(0.025),

Style = TablixStyle.CreateTablixStyle(RdlColor.Black.ToString(), BorderStyle.Dashed.ToString(), 2, System.Drawing.Color.AliceBlue.Name)

};

}

Questo è il primo metodo da inserire in quanto crea l’item del report con relativo stile

CreateTablixColumnHierarchy() ci sono vari overload di questo metodo, il più “completo” :

Genera una gerarchia sulle colonne in base all'elemento passato. inch = distanza tra le colonne. Personalizzabile il valore da visualizzare font = dimensione del font, fontweight = spessore, color = colore, format = formato

public static TablixColumnHierarchy CreateTablixColumnHierarchy(string GroupByElement, string SortByElement, string ShowElement, double inch, int font, FontWeight fontweight, string color, string format) {

var group = new Group(new GroupExpressions(new GroupExpression("")));

if (GroupByElement != null) {

group =

new Group(

new GroupExpressions(new GroupExpression("=" + Expression.FieldsValue(GroupByElement, false))));

}

var sortExpression =

new SortExpression(new Value(""));

if (SortByElement != null) {

sortExpression =

new SortExpression(new Value("=" + Expression.FieldsValue(SortByElement, false)));

}

var sortExpressions = new SortExpressions(sortExpression);

var textRun = new TextRun {

};

if (ShowElement != null) {

textRun = new TextRun {

Value = "=" + Expression.FieldsValue(ShowElement, false),

FontSize = new Point(font),

FontWeight = fontweight,

Color = color,

Format = format

};

}

var paragraph = new Paragraph(new TextRuns(textRun)) { TextAlign = TextAlign.Center };

var textbox = new Textbox(paragraph) { TextboxStyle = new TextboxStyle() };

var header = new TablixHeader(new Inch(inch), new CellContents(textbox));

return new TablixColumnHierarchy(new TablixMembers(new TablixMember(group, sortExpressions, header)));

}

CreateTablixRowHierarchy () ci sono 6 overload di questo metodo ed è speculare al CreateTablixColumnHierarchy

Per quanto riguarda il Body

private static TablixBody CreateTablixBody() {

var tablixRows = new TablixRows();

tablixRows.Add(TablixRow.CreateTablixRow());

var tablixColumns = new TablixColumns(new TablixColumn(new Width(new Inch(1))));

return new TablixBody(tablixColumns, tablixRows);

}

Dove il CreateTablixRow() che ha più overload:

Crea una riga Tablix in una Tablix. Element = campo da visualizzare, subtotal = indica se è subtotale, font = dimensione del font, fontFamily = tipo font,fontweight = spessore, color = colore, format = formato, textAlign = allieamento testo, inch = distanza tra le righe

public static TablixRow CreateTablixRow(string Element, bool subtotal, string fontFamily, int font, FontWeight fontweight, string color, string format, TextAlign textAlign, double inch) {

var textRun = new TextRun {

Value = "=" + Expression.FieldsValue(Element, subtotal),

FontFamily = fontFamily,

FontSize = new Point(font),

FontWeight = fontweight,

Color = color,

Format = format

};

var paragraph = new Paragraph(new TextRuns(textRun)) { TextAlign = textAlign };

var textbox = new Textbox(paragraph) { TextboxStyle = new TextboxStyle() };

var tablixCells = new TablixCells(new TablixCell(new CellContents(textbox)));

return new TablixRow(new Inch(inch), tablixCells);

}

Funzionalità “custom”

È possibile creare TablixRow con più celle (attenzione un tablixRow con 2 celle deve avere 2 colonne sia nella gerarchia che nel var tablixColumns = TablixColumns.CreateColumn(2) nel Body!!)

Si può fare direttamente dall’tablixRows.add

tablixRows.Add(TablixRow.CreateTablixRow(new TablixCell[] {

TablixCell.Create(…),

TablixCell.Create(…)

}));

Oppure

public static TablixRow CreateTablixRow() {

TablixCells tablixCells = new TablixCells();

tablixCells.Add(TablixCell.Create(...));

return new TablixRow(new Inch(0.4), tablixCells);}

Crea una cella da inserire in una row in una tablix. Element = campo da visualizzare, subtotal = indica se è subtotale, fontFamily = tipo font, font = dimensione del font, fontweight = spessore, color = colore, format = formato, textAlign allieamento testo

public static TablixCell Create(string Element, bool subtotal, string fontFamily, int font, FontWeight fontweight, string color, string format,TextAlign textAlign) {

var textRun = new TextRun {

Value = "=" + Expression.FieldsValue(Element, subtotal),

FontFamily = fontFamily,

FontSize = new Point(font),

FontWeight = fontweight,

Color = color,

Format = format,

};

var paragraph = new Paragraph(new TextRuns(textRun)) { TextAlign = textAlign };

var textbox = new Textbox(paragraph) { TextboxStyle = new TextboxStyle() };

return (new TablixCell(new CellContents(textbox)));

}

Oppure si può fare direttamente in maniera pura senza usare metodi facendo:

public static TablixRow CreateTablixRow() {

TablixCells tablixCells = new TablixCells();

var textRun = new TextRun {

Value = "=" + Expression.FieldsValue(ElementProperty.Id2, false),

FontSize = new Point(9),

FontWeight = FontWeight.SemiBold,

Color = RdlColor.Red.ToString(),

};

var paragraph = new Paragraph(new TextRuns(textRun)) { TextAlign = TextAlign.Center };

var textbox = new Textbox(paragraph) { Visibility = new Visibility(true, "Textbox1"), TextboxStyle = new TextboxStyle() };

tablixCells.Add(new TablixCell(new CellContents(textbox)));

textRun = new TextRun {

Value = "=" + Expression.FieldsValue(ElementProperty.Name, false),

FontSize = new Point(9),

FontWeight = FontWeight.Normal,

Color = RdlColor.Black.ToString(),

Format = "n",

};

paragraph = new Paragraph(new TextRuns(textRun)) { TextAlign = TextAlign.Center };

textbox = new Textbox(paragraph) { Visibility = new Visibility(true, "Textbox1"), TextboxStyle = new TextboxStyle() };

tablixCells.Add(new TablixCell(new CellContents(textbox)));

return new TablixRow(new Inch(0.4), tablixCells);

}

**DRILL DOWN**

Per il drilldown è possibile inserire nel metodo:

public static TablixRow CreateTablixRow() {

textbox = new Textbox(paragraph) { Visibility = new Visibility(true, "Textbox1"), TextboxStyle = new TextboxStyle() };

Usando la classe Visibility

Funzione di drilldown. Hidden indica se la cella è visibile o meno, Toggleitem = textbox su cui vale la proprietà

public Visibility(bool Hidden, string Toggleitem) {

this.hidden = Hidden;

this.toggleitem = Toggleitem;

Build();

}

OPPURE Sempre nel CreateTablixRow()

Usare TablixCell.Create(…)

OPPURE DIRETTAMENTE DAL BODY

tablixRows.Add(CreateTablixRow(…)); !!In questo caso la riga conterrà solo 1 cella!!

PARTI FORSE DA TOGLIERE DALLA DOCUMENTAZIONE

Metodi nel dettaglio

private static TablixColumnHierarchy CreateTablixColumnHierarchy() {

var group =

new Group(

new GroupExpressions(new GroupExpression("=" + Expression.FieldsValue(ElementProperty.Id1, false))));

var sortExpression =

new SortExpression(new Value("=" + Expression.FieldsValue(ElementProperty.Id1, false)));

var sortExpressions = new SortExpressions(sortExpression);

var textRun = new TextRun {

Value = "=" + Expression.FieldsValue(ElementProperty.Id1, false),

FontWeight = FontWeight.Default,

Color = RdlColor.Red.ToString()

};

var paragraph = new Paragraph(new TextRuns(textRun)) { TextAlign = TextAlign.Center };

var textbox = new Textbox(paragraph) { TextboxStyle = new TextboxStyle() };

var header = new TablixHeader(new Inch(0.4), new CellContents(textbox));

return new TablixColumnHierarchy(new TablixMembers(new TablixMember(group, sortExpressions, header)));

}

CreateTablixColumnHierarchy serve per inserire nel tablix delle colonne con raggruppamento (var group) per FieldsValue

Se non si volesse fare un raggruppamento si può scrivere come

private static TablixColumnHierarchy CreateTablixColumnHierarchy() {

return new TablixColumnHierarchy(new TablixMembers(new TablixMember()));

}

private static TablixRowHierarchy CreateTablixRowHierarchy() {

var tablixRowMembers = new TablixMembers();

tablixRowMembers.Add(new TablixMember());

tablixRowMembers.Add(new TablixMember());

return new TablixRowHierarchy(tablixRowMembers);

}

Analoga a quella delle colonne crea una gerarchia sulle righe.

Anche questa si può scrivere come

private static TablixRowHierarchy CreateTablixRowHierarchy() {

return new TablixRowHierarchy(new TablixMembers(new TablixMember()));

}

ATTENZIONE il numero di tablixRowMembers.Add(new TablixMember());

deve essere uguale al numero di tablixRows aggiunte nel CreateTablixBody

private static TablixBody CreateTablixBody() {

var tablixRows = new TablixRows();

tablixRows.Add(TablixRow.CreateTablixRow(ElementProperty.Name, false, null, 9, FontWeight.Default, RdlColor.Black.ToString(), "n", TextAlign.Center));

tablixRows.Add(TablixRow.CreateTablixRow(ElementProperty.Name, true,null, 9, FontWeight.Default, RdlColor.Black.ToString(), "n", TextAlign.Center));

var tablixColumns = new TablixColumns(new TablixColumn(new Width(new Inch(1))));

return new TablixBody(tablixColumns, tablixRows);

}

Definisce il corpo della tablix nella quale è possibile specificare le righe della tablix