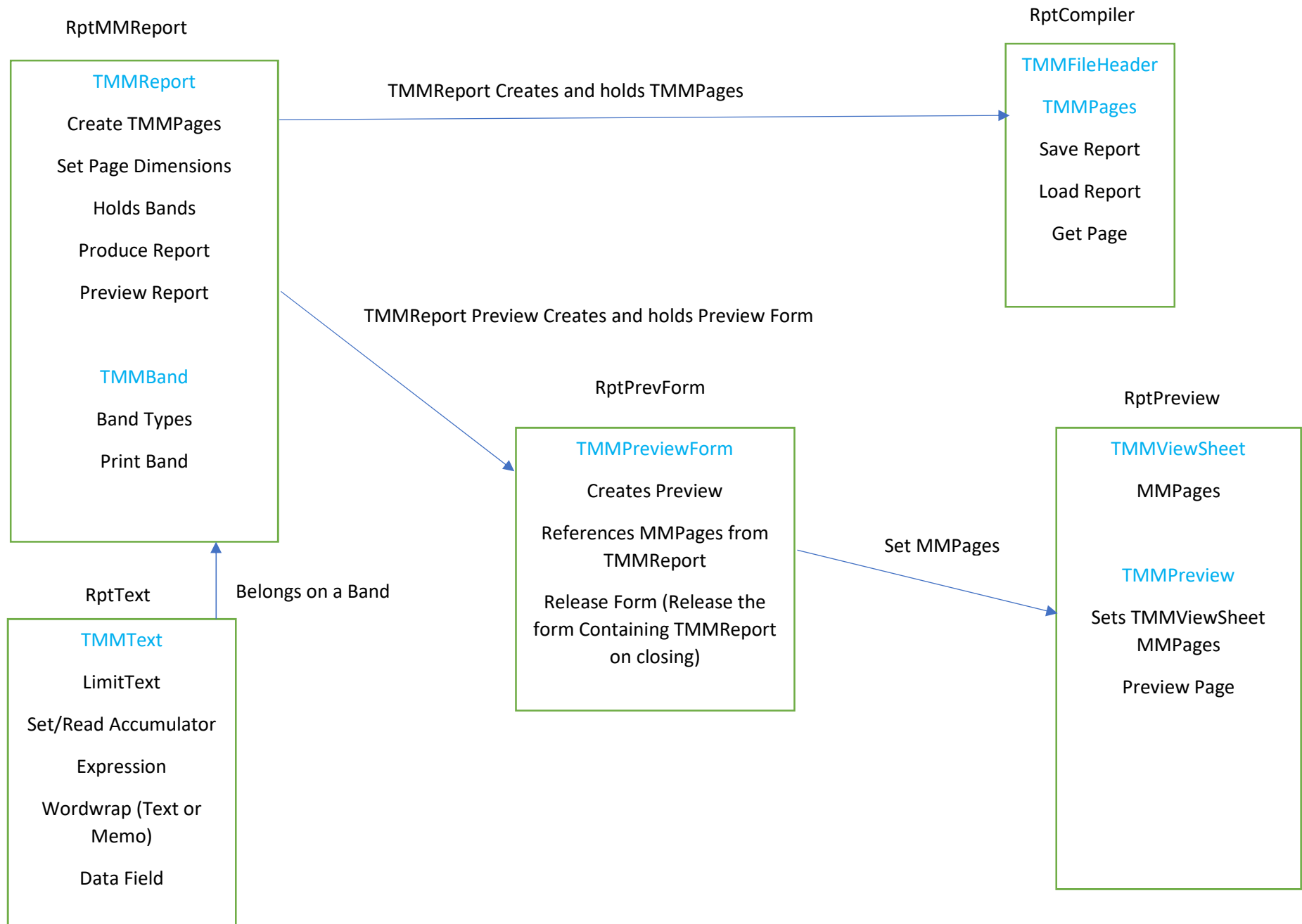


MM Reports Component Reference

© 2018 David McMillan



Normal Application

From an application that requires a report create a Form for the report and from project / Options / Forms take it out of the Auto-create and place it in the Available Forms.

Drop an MMReport on the Form and add bands etc. to make up the report. On that form code create something like a RunReport procedure.

```
procedure TForm2.RunReport;
begin
    MMReport1.PreviewReport(Self); // See FRForm below
End;
```

MMReport1 is the MMReport on the form and Self references the form so that when the preview is closed this form will be released also.

The main program calling this report needs to have the following (assuming the report form is Form2)

```
private
    FReport: TForm2;
```

And to call the report:

```
FReport := TForm2.Create(Self);
    FReport.RunReport;
```

Note that Form2 will get released when the preview is closed – the preview has the following code to do that:

```
procedure TMMPreviewForm.FormClose(Sender: TObject; var Action:
TCloseAction);
begin
    Release; // Releasing the preview on close
    if (FRForm <> nil) then FRForm.Free; // Design Form release
end;
```

To take a more manual approach

```
private
    FReport: TForm2;
    FPreview: TMMPreviewForm;

FReport := TForm2.Create(Self);
FReport.MMReport1.ProduceReport;
FPreview := TMMPreviewForm.Create(Application);
try
    FPreview.MMPages := FReport.MMReport1.MMPages;
    FPreview.ReleaseForm := FReport; // Release FReport on close;
    FPreview.Show;
finally
    // FPreview.Free;
end;
```

Or ShowModal, Don't reference the ReleaseForm and finally free the preview.

Preview from Application without MMReport

Such as when loading a previously saved report. The application needs to create a TMMPages

```
private
    FMMPages: TMMPages;
    FPreview: TMMPreviewForm;

procedure TForm1.FormCreate(Sender: TObject);
begin
    FMMPages := TMMPages.Create;

End;
```

And free it on FormDestroy

```

procedure TForm1.FormDestroy(Sender: TObject);
begin
    FMMPages.Free;
end;

```

Then show the preview with the following in a procedure:

```

FPreview := TMMPreviewForm.Create(Application);
FPreview.MMPPages := FMMPages;
FPreview.Show;

```

The preview will free / release itself on closing it as described above.

Using the Designer

The designer application enables the custom design of reports. Once a report has been designed in the designer it may be accessed from within an application. Further to that the Data Connection can be changed from that in the original report by setting ExternalConnection. The DataSet of the original report can also be replaced by one in the application by setting ExternalDataSource.

Example code:

Uses MMReportRun

```

procedure TForm1.Button1Click(Sender: TObject);
var
    AReport: TMMRptRun;
begin
    AReport := TMMRptRun.Create(Self);
    try
        AReport.ReleaseForm := AReport;
        AReport.ExternalConnection := ADOConnection1;
        AReport.PreviewReport('D:\Temp\Home Database.rpt');
    finally
        // AReport.Free;
    end;
end;

```

Probably can do this without a try... finally because AReport gets released when the ReleaseForm property is set.

In this example the Connection is set to the application (Allows for the same report to be run on different installations)

TMMText

Depending on the settings can be a Text, Memo, show from a database field, an expression, Add the result to an accumulator or read from an accumulator and optionally reset the accumulator to zero after print.

Expressions

#PAGENO #PERIOD For Accounting reports – (uses accumulator 98)

#DATETIME

#PRINTACCUM(2) or any number from 0 to 99 to retrieve a value from the accumulator. AccResetAfterPrint if True will reset the accumulator to zero after printing.

To add to an accumulator on printing set Accumulate to the required Accumulate type

```
TAcMMAccumType = (acNone, acAll, acNegative, acPositive);
```

And indicate the accumulators separated by semicolons

Example 5;7;12;15 or just a single number.

Wordwrap

Default is False. If set to true then there is wordwrap in a memo. If left at false then the text expands to accommodate the data. To limit how much the control can expand LimitText can be set to a value other than zero. Zero allows for any length.

Mask

Set a standard Delphi mask for money, date/time etc.

Alignment

If the text is not wordwrap then alignment will set the text alignment e,g, for money right alignment is generally required.

TMMImage

Has a property Border which if True will print a border around the image. The border details, colour, width... are set by the Pen properties. Half the width ends up under the image but avoids any gap between the border and the image.

Source File Locations

Components and preview are held in

C:\Program Files (x86)\Embarcadero\Studio\19.0\LibMMReport also

1. RptDesignc – the component design grabs in the report designer.
2. MMReportRun – The unit that is accessed to run a predesigned report and preview it.
3. FDReportRun – FireDac run for SQLite

D:\Delphi 10\MMReport

Has

1. the Report Writer source,
2. Application Example. An example of creating a report within an application and previewing it
3. Run Example. An example of running a report written in the report designer.

Sub Detail Band

Any Sub Detail band created will not print until the following happens

1. The Sub Detail band is given a distinct name
2. procedure **MMReport.SetSubToPrint(Value: String; aRepeat: Boolean);** is used to set the name of the Sub Detail to print.

If the name of a **Sub** Detail is set to print (By SetSubToPrint) then the band will always print after the **Detail** band – that however is not particularly useful. The idea is to set the **Sub** Detail band to print from within the **Detail** Band BeforePrint – there the **Detail** Band DoPrint can be set if required as well. It may be that the next line needs a different **Sub** Detail band. For each **Sub** Detail band required (or none) the **Detail** Band BeforePrint needs to access SetSubToPrint. These values can be set programmatically or from a list of instructions as in an accounts report for example. (Perhaps just use a global variable for the name that gets changed). From within the **Detail** Band BeforePrint data values can be accessed to decide which **Sub** Detail to print if needed.

Unlike other band types of which only the first is used (i.e. only one of each type is permitted) there is no set limit to the number of Sub Detail bands which allows for Accounts reports to print different bands according to an instruction list.

MMReport.SetSubToPrint(Value: String; aRepeat: Boolean);

If aRepeat is true then the Sub Detail band will repeat indefinitely. One way to end the repeats is to make a change in the Sub Detail BeforePrint or AfterPrint.

So for example in the **Detail Band AfterPrint**

If sometest then MMReport1.SetSubToPrint('MMBand5',True);

And in the **Sub Detail Band BeforePrint**

Move DataSet or update counter or whatever

if Sometest then

begin

MMReport1.SetSubToPrint('MMBand5',False);

Reset sometest;

DoPrint := False;

end;

Another way is to use the **property OnDataMove** event For the MMReport. This event is triggered immediately after the data is moved so that a list of instructions can be used to change which Sub Detail to print or test before printing.