Creating a Basic PivotTable

Table of Contents

Intro

⚠ The ability to manipulate PivotTables in ExcelApplication was introduced in OfficeWriter 8.4.0.

⚠ This example assumes a basic level of understanding the different parts of a PivotTable and how they work. If you are not familiar with PivotTable terminology, we highly recommend that you read this Special Topics article before continuing.

This example takes an existing workbook that contains some data and creates a PivotTable. The workbook used in this example is available for download: Download BasicExample.xlsx.

Before writing any PivotTable code, make sure to open the workbook with `ExcelApplication` and get references to the data worksheet and a worksheet for the PivotTable. See Adding OfficeWriter to your .NET Application.

```
ExcelApplication xla = new ExcelApplication();

//Example.xlsx has a worksheet 'Data' with 9 columns and 244 rows of data
Workbook wb = xla.Open("\\BasicExample.xlsx");

Worksheet data_ws = wb.Worksheets["Data"];
Worksheet pivot_ws = wb.Worksheets["Pivot"];
```

This example places the PivotTable on a separate sheet. It is recommended that each PivotTable be placed on a separate sheet.

Excel does not allow PivotTables to overlap and will use error messages to prevent users from creating overlapping PivotTables. ExcelWriter does not have the ability to render PivotTables, so it cannot detect if two PivotTables will overlap when they are rendered. To avoid this, we encourage you to keep your PivotTables on separate worksheets.

Writing the code

Step 1. Set up the data source

The data source needs to be a continuous block of cells with a header row with column names. The data source is defined as an `Area`.

Here is a snapshot of the data for this tutorial, which can also be found in **BasicExample.xlsx**:

	A	B	C	D	E	F	G	H	I
1	ProductSubCategory Name	Product Name	Sales	Special Offer Start Date	Special Offer End Date	Product Standard Cost	Product List Price	Special Offer Dicount	Unit Price with Discount
2	Pedals	HL Mountain Pedal	Half-Price Pedal Sale	11/21/2012	11/26/2012	35.96	80.99	0.5	40.5
3	Pedals	LL Road Pedal	Half-Price Pedal Sale	11/21/2012	11/26/2012	17.98	40.49	0.5	20.25
4	Pedals	ML Road Pedal	Half-Price Pedal Sale	11/21/2012	11/26/2012	27.57	62.09	0.5	31.05
5	Pedals	HL Road Pedal	Half-Price Pedal Sale	11/21/2012	11/26/2012	35.96	80.99	0.5	40.5
6	Pedals	Touring Pedal	Half-Price Pedal Sale	11/21/2012	11/26/2012	35.96	80.99	0.5	40.5
7	Pedals	LL Mountain Pedal	Half-Price Pedal Sale	11/21/2012	11/26/2012	17.98	40.49	0.5	20.25
8	Pedals	ML Mountain Pedal	Half-Price Pedal Sale	11/21/2012	11/26/2012	27.57	62.09	0.5	31.05
9	Road Frames	LL Road Frame - Black 58	LL Road Frame Sale	12/5/2012	12/12/2012	204.63	337.22	0.35	219.19
10	Road Frames	LL Road Frame - Black 60	LL Road Frame Sale	12/5/2012	12/12/2012	204.63	337.22	0.35	219.19
11	Road Frames	LL Road Frame - Black 62	LL Road Frame Sale	12/5/2012	12/12/2012	204.63	337.22	0.35	219.19
12	Road Frames	LL Road Frame - Red 44	LL Road Frame Sale	12/5/2012	12/12/2012	187.16	337.22	0.35	219.19
13	Road Frames	LL Road Frame - Red 48	LL Road Frame Sale	12/5/2012	12/12/2012	187.16	337.22	0.35	219.19
14	Road Frames	LL Road Frame - Red 52	LL Road Frame Sale	12/5/2012	12/12/2012	187.16	337.22	0.35	219.19
15	Road Frames	LL Road Frame - Red 58	LL Road Frame Sale	12/5/2012	12/12/2012	187.16	337.22	0.35	219.19
16	Road Frames	LL Road Frame - Red 60	LL Road Frame Sale	12/5/2012	12/12/2012	187.16	337.22	0.35	219.19
17	Road Frames	LL Road Frame - Red 62	LL Road Frame Sale	12/5/2012	12/12/2012	187.16	337.22	0.35	219.19
18	Road Frames	LL Road Frame - Black 44	LL Road Frame Sale	12/5/2012	12/12/2012	204.63	337.22	0.35	219.19
19	Road Frames	LL Road Frame - Black 48	LL Road Frame Sale	12/5/2012	12/12/2012	204.63	337.22	0.35	219.19
20	Road Frames	LL Road Frame - Black 52	LL Road Frame Sale	12/5/2012	12/12/2012	204.63	337.22	0.35	219.19
21	Tires and Tubes	LL Mountain Tire	Mountain Tire Sale	1/17/2012	1/19/2012	9.35	24.99	0.5	12.5

There are 9 columns and 244 rows in the data set, including the row with the header values.

In this case, the data source for the PivotTable will be a dynamically defined area on the data worksheet. Note that the row of column names is included in the area.

```
Area data_area = data_ws.CreateArea(0, 0, 244, 9);
```

Step 2. Create the PivotTable

To create a `PivotTable`, call `CreatePivotTable` on the `PivotTables` collection. Specify the 0-indexed row and column values for the PivotTable location:

```
PivotTable pt = pivot_ws.PivotTables.CreatePivotTable(data_area, 0, 0);
```

This example shows a PivotTable being created from a fully populated data set. PivotTables can also be created and modified in files that will be populated by `ExcelTemplate`. The same principles apply whether the data source contains values or data markers.

Step 3. Set PivotTable Options

In Excel, there are a number of options that can be set by going to *PivotTable Options*. These properties are available through `PivotTableSettings`.

There are a couple of properties that you should always consider when working with PivotTables with ExcelWriter.

Refresh On Open

After creating the PivotTable, always set `RefreshOnOpen` to `true`.

```
pt.PivotTableSettings.RefreshOnOpen = true;
```

ExcelWriter does not have the ability to render a PivotTable, so any modifications made to a PivotTable will not take affect until the output file is opened in Excel and the PivotTable is refreshed. If `RefreshOnOpen` is `true`, Excel will refresh the PivotTable when the workbook opens, which will re-render the PivotTable.

ItemsToRetain

The other important property to set is `ItemsToRetain`. When the PivotTable is created, the values that are available through the row label, column label, or page field drop-down filters are based on the values in the data source of the PivotTable at the time the PivotTable was created.

By default, the PivotTable will retain all the original values in those filters, even if those values are no longer in the data source. Set `ItemsToRetain` to `None` to make sure the original values are cleared out when the PivotTable is refreshed.

```
pt.PivotTableSettings.ItemsToRetain.Value = ItemsToRetain.None;
```


This is especially important when working with PivotTables that have data markers in the data source. Set `ItemsToRetain` to keep data markers out of the filters in the final output.

Step 4. Add PivotTableFields

There are four types of `PivotTableField`: `DataFields`, `RowLabels`, `ColumnLabels`, and `PageFields` (also called report filters). `PivotTableFields` are created from read-only `SourceFields`, which are generated based on the `PivotTable` data source.

SourceFields

A `SourceField` can be used to create one `PageField`, `ColumnLabel`, or `RowLabel`, in addition to an unlimited number of `DataFields`.

Get a handle on the `SourceFields` to use for building `PivotTableFields` later. `SourceFields` can be referenced by the column header name or by column index.

```
SourceField prodSubCategory = pt.SourceFields["ProductSubCategoryName"];
SourceField prodName = pt.SourceFields["Product Name"];
SourceField sales = pt.SourceFields["Sales"];
SourceField dateStart = pt.SourceFields["Special Offer Start Date"];
SourceField dateEnd = pt.SourceFields["Special Offer End Date"];
SourceField prodStdCost = pt.SourceFields["Product Standard Cost"];
SourceField listPrice = pt.SourceFields["Product List Price"];
SourceField discount = pt.SourceFields["Special Offer Dicount"];
SourceField unitDiscountPrice = pt.SourceFields["Unit Price with Discount"];
```

PageFields

To add a `PageField`, call `CreateField` on the `PivotTable.PageFields` collection. You will need to specify the `SourceField` that will be used to create the `PageField`.

```
pivot.PageFields.CreateField(dateStart);
pivot.PageFields.CreateField(dateEnd);
```

The layout of the page fields depends on whether the page layout is set to 'Down, then Over' or 'Over, then Down' and the number of fields allowed per row/column. These properties are not exposed in the current API, so the defaults apply when creating a PivotTable from scratch. The defaults are 'Down, then Over' and an unlimited number of fields per column.

This means that in PivotTables created by ExcelWriter, page fields are always placed starting two rows above the upper left corner of the PivotTable, so if a PivotTable is placed in cell B10, then page fields will start in B8. Any additional page fields are added to B8 and existing page fields are moved up, or the new page field is added to a column to the right, depending on the page layout properties in the PivotTable.

By default, all the page fields appear in one column.

If there is no space to add the page fields above the PivotTable (i.e. the PivotTable is located in cell A1), then ExcelWriter will automatically move the PivotTable down to accommodate the page fields.

RowLabels and ColumnLabels

Similarly to page fields, `RowLabels` and `ColumnLabels` are created on the `PivotTable.RowLabels` and `PivotTable.ColumnLabels` collections. As mentioned earlier, only one `RowLabel`, `ColumnLabel`, or `PageField` can be created from a particular `SourceField`.

```
PivotTableField prodSubCategoryFld = pt.RowLabels.CreateField(prodSubCategory);
PivotTableField prodNameFld = pt.RowLabels.CreateField(prodName);
PivotTableField salesFld = pt.RowLabels.CreateField(sales);
```

Row labels and column labels display the subtotals for each group of values. The subtotal can be displayed at the top or bottom of the group.

```
prodNameFld.DisplaySubtotalsAtTop = true;
salesFld.DisplaySubtotalsAtTop = false;
```

Excel automatically sorts and re-renders a PivotTable any time a change is made. By default, the row label or column label values are sorted alpha-numerically in ascending order.

Since ExcelWriter does not have the ability to render PivotTables or sort the values for a field, the only way to guarantee that the data will be sorted is to set `RefreshOnOpen` to `true` and set `SortOptions.Ordering` on a `PivotTableField` to be `Ascending` or `Descending`.

This property only affects row and column labels.

```
prodSubCategoryFld.SortOptions.Ordering = SortOptions.OrderingType.Ascending;
prodNameFld.SortOptions.Ordering = SortOptions.OrderingType.Ascending;
salesFld.SortOptions.Ordering = SortOptions.OrderingType.Ascending;
```

When Excel refreshes the PivotTable, it will observe the `SortOptions` setting for a particular field.

DataFields

To create a data field, call `CreateField` on the `DataFields` collection. Unlike row labels, column labels, and page fields, multiple data fields can be created from the same source field.

```
PivotTableField lp = pt.DataFields.CreateField(listPrice);
PivotTableField dct = pt.DataFields.CreateField(discount);
PivotTableField udct = pt.DataFields.CreateField(unitDiscountPrice);
```

In Excel, a unique name is given to the data field depending on the type of data in the source field (numerical or mixed), whether or not any other data fields were already created from the same source field, and whether the source field name ends in a number or alphabetical character (e.g. Case1 vs. CaseOne).

ExcelWriter uses a consistent naming convention when creating data fields: all data fields follow the format *SOURCEFIELDNAME#, where _#* is an incremental number starting at 1. To change the name of a data field, use the `DisplayName` property.

```
lp.DisplayName = "Avg List Price";
dct.DisplayName = "Avg Discount";
udct.DisplayName = "Avg Discount Price";
```

The function used to aggregate the data can be specified through the *summarize by* property. If a column data contains just numbers, the function defaults to SUM. If the column contains mixed data, the function defaults to COUNT. There are other functions available.

```
lp.SummarizeBy = PivotTableField.SummarizeByType.Average;
dct.SummarizeBy = PivotTableField.SummarizeByType.Average;
udct.SummarizeBy = PivotTableField.SummarizeByType.Average;
```

A number format can be specified for all PivotTable fields, but it will only take affect on data fields and row labels/column labels/page fields that

have numeric data only.

```
lp.NumberFormat = "$#,##0.00";
dct.NumberFormat = "0.00%";
udct.NumberFormat = "$#,##0.00";
```

Completed Code

And that concludes how to create a basic PivotTable. Here is the full sample code below:

```
ExcelApplication xla = new ExcelApplication();
Workbook wb = xla.Open("\\BasicExample.xlsx"); //Template already contains the data

Worksheet data_ws = wb.Worksheets["Data"];
Worksheet pivot_ws = wb.Worksheets["Pivot"];

Area data_area = data_ws.CreateArea(0, 0, 244, 9);

//Create a PivotTable in cell A1 on worksheet 'Pivot'
PivotTable pt = pivot_ws.PivotTables.CreatePivotTable(data_area, 0, 0);

//Set some of the PivotTable options
pt.PivotTableSettings.RefreshOnOpen = true;
pt.PivotTableSettings.ItemsToRetain.Value = ItemsToRetain.None;

//Get a handle on the source fields
SourceField prodSubCategory = pt.SourceFields["ProductSubCategoryName"];
SourceField prodName = pt.SourceFields["Product Name"];
SourceField sales = pt.SourceFields["Sales"];
SourceField dateStart = pt.SourceFields["Special Offer Start Date"];
SourceField dateEnd = pt.SourceFields["Special Offer End Date"];
SourceField prodStdCost = pt.SourceFields["Product Standard Cost"];
SourceField listPrice = pt.SourceFields["Product List Price"];
SourceField discount = pt.SourceFields["Special Offer Dicount"];
SourceField unitDiscountPrice = pt.SourceFields["Unit Price with Discount"];

//Create some page fields
pt.PageFields.CreateField(dateStart);
pt.PageFields.CreateField(dateEnd);

//Create some row labels
PivotTableField prodSubCategoryFld = pt.RowLabels.CreateField(prodSubCategory);
PivotTableField prodNameFld = pt.RowLabels.CreateField(prodName);
PivotTableField salesFld = pt.RowLabels.CreateField(sales);

//Set where the subtotals for the groups are displayed
prodNameFld.DisplaySubtotalsAtTop = true;
salesFld.DisplaySubtotalsAtTop = false;

//Set the sort type to be ascending
prodSubCategoryFld.SortOptions.Ordering = SortOptions.OrderingType.Ascending;
prodNameFld.SortOptions.Ordering = SortOptions.OrderingType.Ascending;
salesFld.SortOptions.Ordering = SortOptions.OrderingType.Ascending;

//Create some data fields
PivotTableField lp = pt.DataFields.CreateField(listPrice);
```

```
PivotTableField dct = pt.DataFields.CreateField(discount);
PivotTableField udct = pt.DataFields.CreateField(unitDiscountPrice);

//Set the display name
lp.DisplayName = "Avg List Price";
dct.DisplayName = "Avg Discount";
udct.DisplayName = "Avg Discount Price";

//Set the summarize by values
lp.SummarizeBy = PivotTableField.SummarizeByType.Average;
dct.SummarizeBy = PivotTableField.SummarizeByType.Average;
udct.SummarizeBy = PivotTableField.SummarizeByType.Average;

//Set the number format
lp.NumberFormat = "$#,##0.00";
dct.NumberFormat = "0.00%";
udct.NumberFormat = "$#,##0.00";
```

```
xla.Save(wb, "\\BasicExampleOut.xlsx");
```

Additional Resources

- Templates and PivotTables - using PivotTables with ExcelTemplate
- Intro to PivotTables - Terminology from Excel