Sales Commission Report

Intro

An example of a report created with ExcelWriter that compares previous year and year-to-date sales data for a sales team.

The source Workbook contains a listing of Employees for the fictional company AdventureWorks. The actual data is marked with two Named Ranges, "EmployeesHeadings" and "EmployeesData", which will make it easier to import using ExcelWriter. This data is then used to populate a DataGrid, which will be displayed below.

Code

```
public class Commission
   {
       private ExcelApplication xlw;
       private Workbook wb;
       private Worksheet ws, wsChart;
       private Area importedArea;
       private bool ChartsYtd;
       private string chartFontName = "Garamond";
       private Color clrChartArea, clrLegend;

       /// <summary>
       /// Build the report with ExcelApplication
       /// </summary>
       public void GenerateReport(bool ytd)
       {
           //If ChartsYtd the pie chart will show year to date data
           ChartsYtd = ytd;

           // Create an instance of ExcelApplication
           //and open the template file.

           xlw = new ExcelApplication();

           // Open the template workbook
           string templatePath = @"..\..\ExcelTemplateFiles\SalesCommission.xlsx";
           wb = xlw.Open(templatePath);

           ws = wb.Worksheets[0];

           // Setup the colors that will be used in the workbook
           this.SetupColors();

           // Import the data into the worksheet
           this.PopulateWorksheet();

           // Create a new worksheet to hold the charts.
           //Set the gridline visibilty and tab color.

           wsChart = wb.Worksheets.CreateWorksheet("Charts");
           wsChart.ShowGridlines = false;
           wsChart.TabColor = clrLegend;

           // Add the charts
           this.AddPieChart();
```

```
            this.AddColumnChart();

            // Save the report by streaming
             //it to the client's browser
            string reportName;
            if (this.ChartsYtd)
            {
                reportName = "SalesCommissionReport-YTD.xlsx";
            }
            else
            {
                reportName = "SalesCommissionReport-LastYear.xlsx";
            }
            xlw.Save(wb, @"..\..\ExcelOutputFiles\"+reportName);
        }

        /// <summary> Initialize the colors to be used in
        /// the workbook.  These colors will be applied
        /// to the chart areas and chart legends.
        /// </summary>
        private void SetupColors()
        {
            // Get the colors from the Workbook Palette object
            Palette pal = wb.Palette;
            clrChartArea = pal.GetClosestColor(240, 243, 248);
            clrLegend = pal.GetClosestColor(234, 234, 234);
        }

        /// <summary> Add the column chart to the worksheet.</summary>
        private void AddColumnChart()
        {
            // This chart will sit near cell A25
            Anchor anch = wsChart.CreateAnchor(24, 0, 50, 50);

            // Create the chart
            Chart colChart = wsChart.Charts.CreateChart(ChartType.Column.Clustered3D,
anch);

            // Set title, title font, and shadow settings
            colChart.ChartArea.HasShadow = true;
            colChart.Title.Text = "Commission Payout Comparison";
            colChart.Title.Font.Name = chartFontName;
            colChart.Title.Font.Size = 22;
            colChart.Legend.Font.Name = chartFontName;

            // Set the chart area and legend colors.
            //These colors were initialized in the setupColors() method.

            colChart.ChartArea.Interior.ForegroundColor = clrChartArea;
            colChart.Legend.Interior.ForegroundColor = clrLegend;

            // Create areas for the chart's category data, and for the
            //YTD and last year series objects.

            Area categoryArea = ws.CreateArea(6, 1, importedArea.RowCount, 1);
            Area seriesAreaLast = ws.CreateArea(6, 5, importedArea.RowCount, 1);
            Area seriesAreaYTD = ws.CreateArea(6, 4, importedArea.RowCount, 1);

            // Set category data. This will be the employee names on the X axis.
```

```
        colChart.SeriesCollection.CategoryData = categoryArea.Dimensions;

        // Create two series objects.  Last Year and YTD will be displayed
        //as chart columns side-by-side.

        Series srsLast = colChart.SeriesCollection.CreateSeries(seriesAreaLast);
        srsLast.Name = "Last Year";
        Series srsYTD = colChart.SeriesCollection.CreateSeries(seriesAreaYTD);
        srsYTD.Name = "YTD";
    }

    /// <summary> Add a pie chart to the chart worksheet.</summary>
    private void AddPieChart()
    {
        // This chart will sit near cell A1
        Anchor anch = wsChart.CreateAnchor(0, 0, 50, 50);

        // Create a Pie3D chart
        Chart pieChart = wsChart.Charts.CreateChart(ChartType.Pie.Pie3D, anch);

        pieChart.ChartArea.HasShadow = true;

        // Determine whether to show YTD or Last Year data
        //based on the value passed in by the user.

        int commCol;
        if (ChartsYtd)
        {
            // YTD data is in the 5th column.
            commCol = 4;
            pieChart.Title.Text = "Commission Distribution - YTD";
            wsChart.Name = "Charts - YTD";
        }
        else
        {
            // Last Year data is in the 6th column.
            commCol = 5;
            pieChart.Title.Text = "Commission Distribution - Last Year";
            wsChart.Name = "Charts - Last Year";
        }

        // Set Chart title font appearance
        pieChart.Title.Font.Name = chartFontName;
        pieChart.Title.Font.Size = 22;
        pieChart.Legend.Font.Name = chartFontName;

        // Set chart area and legend shading.
        //These colors were initialized in the setupColors() method.

        pieChart.ChartArea.Interior.ForegroundColor = clrChartArea;
        pieChart.Legend.Interior.ForegroundColor = clrLegend;

        // Create Area objects for category and series data
        Area categoryArea = ws.CreateArea(6, 1, importedArea.RowCount, 1);
        Area seriesArea = ws.CreateArea(6, commCol, importedArea.RowCount, 1);

        // Set the category and series data
        pieChart.SeriesCollection.CategoryData = categoryArea.Dimensions;
        Series commSeries = pieChart.SeriesCollection.CreateSeries(seriesArea);
```

```
            // Show chart leader lines and values as percentages.
            commSeries.SettingsPieDoughnut.ShowLeaderLines = true;
            commSeries.DataLabels.ContainsValueAsPercentage = true;
            commSeries.DataLabels.Font.Name = chartFontName;
            commSeries.Line.Visible = true;
        }

        /// <summary> Import the data from a 2-D Object array
        /// into the worksheet.
        /// </summary>
        private void PopulateWorksheet()
        {
            DataImportProperties dp = wb.CreateDataImportProperties();
            dp.Truncate = true;

            // The workbook has a named range "DataRange"
            //defined that marks where the data should be imported.
            //Get a reference to this range.

            Range dataRange = wb.GetNamedRange("DataRange");
            Area dataArea = dataRange.Areas[0];

            // These arrays are the data to import
            object[,] data = this.GetData();
            string[] fieldNames = { "Name", "SalesYTD", "SalesLastYear" };

            // Import the data into the data area.
            importedArea = dataArea.ImportData(data, fieldNames, dp);

            // Remove extra rows from the worksheet
            int lastDataAreaRow = dataArea.FirstRow + dataArea.RowCount - 1;
            int lastImportedAreaRow = importedArea.FirstRow + importedArea.RowCount -
1;
            while (lastDataAreaRow > lastImportedAreaRow)
            {
                ws.DeleteRow(lastDataAreaRow);
                --lastDataAreaRow;
            }

        }

        /// <summary> ExcelWriter can import data from arrays as well as DataTables.
        /// The data returned by this method will be imported into the worksheet.
        /// </summary>
        /// <returns> 2-D Object array of data
        /// </returns>
        private object[,] GetData()
        {
            object[,] data = {{"Ansman-Wolfe, Pamela", 2488342.5141, 1927059.1780},
                              {"Blythe, Michael", 2590055.1774, 1750406.4785},
                              {"Campbell, David", 1870183.5288, 1371635.3158},
                              {"Caro, Fernando", 1958815.8056, 1997186.2037},
                              {"Ito, Shu", 1679586.6629, 2073505.9999},
                              {"Mitchell, Linda", 2343461.0492, 1439156.0291},
                              {"Pak, Jae", 2568244.0549, 1635823.3967},
                              {"Reiter, Tsvi", 1855106.2631, 1849640.9418},
                              {"Saraiva, Jos?", 2153295.1978, 2038234.6549},
                              {"Tsoflias, Lynn", 2486869.8048, 2278548.9776},
```

```
                            {"Valdez, Rachel", 2160347.3087, 1307949.7917},
                            {"Vargar, Garrett", 2088272.9112, 1620276.8966},
                            {"Varkey Chudukatil, Ranjit", 2177055.6488,
2396539.7601}};
            return data;
```

```
        }
    }
```

Downloads

- Template file - SalesCommission.xlsx
- Output file for YTD - SalesCommissionReport-YTD.xlsx
- Output file for Last Year - SalesCommissionReport-LastYear.xlsx