Template to Application Sample

Intro

Populate a template with ExcelTemplate and then programmatically manipulate or 'post process' the workbook with ExcelApplication.

This demo begins with an ExcelTemplate workbook with data markers. First, the ExcelTemplate object binds data to the data markers while preserving all existing Excel formats that were in the original template. Then, the populated workbook is opened as an ExcelApplication Workbook where it can be programmatically manipulated. ExcelApplication adds a chart.

Code

```
public class TempToApp
    {
        // Both ExcelApplication and ExcelTemplate
        //objects will be used for this demo.

        private ExcelTemplate xlt;
        private ExcelApplication xlw;
        private Workbook wb;

        /// <summary>
        /// Build the report with ExcelApplication
        /// </summary>
        public void GenerateReport()
        {

            PopulateTemplate();
            AddChart();

            // Save the report to specified folder
            xlw.Save(wb, @"..\..\ExcelOutputFiles\TempToApp_output.xlsx");
        }

        /// <summary> Add a column chart to the second worksheet using
ExcelApplication.
        /// The chart will show data imported with the ExcelTemplate
        /// object.</summary>
        private void AddChart()
        {
            // Get the first two worksheets and give them names
            Worksheet ws = wb.Worksheets[0];
            ws.Name = "Data";

            Worksheet ws2 = wb.Worksheets.CreateWorksheet("ChartSheet");

            // Create a chart on the second worksheet
            Anchor anch = ws2.CreateAnchor(0, 0, 50, 50);
            Chart chrt = ws2.Charts.CreateChart(ChartType.Column.Clustered, anch);

            // Set series and category data
            Series srs1 = chrt.SeriesCollection.CreateSeries("=Data!B2:B7");
            chrt.SeriesCollection.CategoryData = "=Data!A2:A7";
            srs1.NameFormula = "=Data!A1";

            // Configure the chart's legend
```

```
            Legend lgnd = chrt.Legend;
            lgnd.Visible = true;
            lgnd.Location = Legend.LegendLocation.Right;

            // Set the chart's Title string and display properties.
            chrt.Title.Text = "AdventureWorks Global Sales";
        }

        /// <summary> Populate the template workbook with database
        /// data using the ExcelTemplate object.  The ExcelTemplate
        /// save() method returns a Workbook.
        /// </summary>
        private void PopulateTemplate()
        {
            // Create an instance of ExcelTemplate and open
            //the template workbook

            xlt = new ExcelTemplate();
            xlt.PreserveStrings = false;
            // Open the template workbook
            string templatePath =
@"..\..\ExcelTemplateFiles\TemplateToAppTemplate.xlsx";
            xlt.Open(templatePath);

            // Programmatically contruct datatable
            DataTable dt = new DataTable();
            for (int i = 0; i < 3; i++)
                dt.Columns.Add();
            dt.Rows.Add(new string[] {"AU","$12,197,515.53"});
            dt.Rows.Add(new string[] {"CA","$21,515,540.46"});
            dt.Rows.Add(new string[] {"DE","$5,939,763.50"});
            dt.Rows.Add(new string[] {"FR","$9,136,704.47"});
            dt.Rows.Add(new string[] {"GB","$9,506,447.59"});
            dt.Rows.Add(new string[] {"US","$85,387,883.82"});

            xlt.BindData(dt, "Sales", xlt.CreateDataBindingProperties());
            xlt.Process();

            // Here the populated ExcelTemplate object
            //is being opened as an ExcelApplication Workbook.
            //The object can now be programatically manipulated
            //with the ExcelApplication API

            xlw = new ExcelApplication();
            wb = xlw.Open(xlt);
        }
```

```
    }
```

Downloads

- Sample Template File: TemplateToAppTemplate.xlsx
- Sample Output File: TempToApp_output.xlsx