Passing ExcelTemplate to ExcelApplication

ExcelWriter allows you to generate a spreadsheet from script alone - using the ExcelApplication object - or from a template spreadsheet and a script, using ExcelTemplate. ExcelTemplate provides an intuitive high-performance way to import database values to a spreadsheet, but cannot otherwise modify a spreadsheet at runtime. ExcelApplication's rich object model allows you to modify every aspect of the spreadsheet at runtime. You can take advantage of the features of both ExcelApplication and ExcelTemplate by using them together. For example, you can use ExcelTemplate to open and populate an ExcelWriter template, then pass the populated workbook to ExcelApplication and add a chart.
To pass a workbook from ExcelTemplate to ExcelApplication, do not call ExcelTemplate.Save. Instead, pass the ExcelTemplate object to ExcelApplication's Open method:

```
ExcelTemplate xlt = new ExcelTemplate();
xlt.Open(templatePath);
xlt.BindData(data, "Sales", xlt.CreateDataBindingProperties());
xlt.Process();

//--- Create an instance of ExcelApplication and
//--- open the spreadsheet you created with ExcelTemplate.
//--- The spreadsheet will be returned as a Workbook
//--- object.
ExcelApplication xla = new ExcelApplication();
Workbook wb = xla.Open(xlt);
```

Code Sample: Passing ExcelTemplate to ExcelApplication

```
//-----------------------------------------------------------------
//--- SoftArtisans OfficeWriter ExcelApplication Template-To-App Sample
//---
//--- Demonstrates how the ExcelTemplate and ExcelApplication objects
//--- can be used together in the same request.
//--- Populate an ExcelTemplate workbook with data and then open the
//--- populated workbook with the ExcelApplication API for further edits.
//---
//--- (c) 2009 SoftArtisans, Inc.
//--- Support: http://support.softartisans.com
//--- Sales:   sales@softartisans.com
//-----------------------------------------------------------------

using System;
using System.Data;
using System.Data.SqlClient;
using System.Configuration;
using SoftArtisans.OfficeWriter.ExcelWriter;

namespace SoftArtisans.OfficeWriter.ExcelWriter.Samples
{
    /// <summary>
    /// Summary description for TemplateToApp.
    /// </summary>
    public class TemplateToApp : System.Web.UI.Page
    {
        /* Both ExcelApplication and ExcelTemplate
        * objects will be used for this demo.
        */
```

```
        private ExcelTemplate xlt;
        private ExcelApplication xlw;
        private Workbook wb;
        protected System.Web.UI.HtmlControls.HtmlInputButton btnTemplateToApp;
        private string connString =
            System.Configuration.ConfigurationManager.AppSettings["connString"];


        private void btnTemplateToApp_ServerClick(object sender, System.EventArgs e)
        {
            GenerateReport();
        }

        /// <summary>
        /// Build the report with ExcelApplication
        /// </summary>
        private void GenerateReport()
        {
            PopulateTemplate();
            AddChart();

            /* Save the report by streaming it
             * to the client's browser */
            xlw.Save(wb, Page.Response, "TemplateToApp.xls", false);
        }

        /// <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();

            /* Open the template workbook */
            string templatePath = Page.MapPath("templates/TemplateToAppTemplate.xls");
            xlt.Open(templatePath);

            /* Get the data from the database and set it as
            * a data source.
            */
            DataTable dt = GetData();
            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);
        }

        /// <summary> Query the database for the data that will
        /// be imported by the ExcelTemplate object.
        /// </summary>
        /// <returns>DataTable of data for the report</returns>
        private DataTable GetData()
        {
            string sql = "SELECT Person.StateProvince.CountryRegionCode As Country, "
+
                "SUM(Sales.SalesOrderHeader.TotalDue) As TotalSales " +
                "FROM Sales.SalesOrderHeader " +
                "JOIN Sales.CustomerAddress " +
                "ON Sales.SalesOrderHeader.CustomerID=Sales.CustomerAddress.CustomerID
" +
                "JOIN Person.Address " +
                "ON Sales.CustomerAddress.AddressID=Person.Address.AddressID " +
                "JOIN Person.StateProvince " +
                "ON
Person.StateProvince.StateProvinceID=Person.Address.StateProvinceID " +
                "GROUP BY Person.StateProvince.CountryRegionCode";

            DataTable dt = new DataTable();
            using(SqlConnection conn = new SqlConnection(connString))
                new SqlDataAdapter(sql, conn).Fill(dt);

            return dt;
        }

        #region Web Form Designer generated code
        override protected void OnInit(EventArgs e)
        {
            //
```

```
            // CODEGEN: This call is required by the ASP.NET Web Form Designer.
            //
            InitializeComponent();
            base.OnInit(e);
        }

        /// <summary>
        /// Required method for Designer support - do not modify
        /// the contents of this method with the code editor.
        /// </summary>
        private void InitializeComponent()
        {
            this.btnTemplateToApp.ServerClick += new
System.EventHandler(this.btnTemplateToApp_ServerClick);

        }
        #endregion
```

```
    }
}
```

```
<p>Option Strict On<br />
Option Explicit On</p>


<p>&apos;-----------------------------------------------------------------<br />
&apos;--- SoftArtisans OfficeWriter ExcelApplication Template-To-App Sample<br />
&apos;---<br />
&apos;--- Demonstrates how the ExcelTemplate and ExcelApplication objects<br />
&apos;--- can be used together in the same request.<br />
&apos;--- Populate an ExcelTemplate workbook with data and then open the<br />
&apos;--- populated workbook with the ExcelApplication API for further edits.<br />
&apos;---<br />
&apos;--- (c) 2009 SoftArtisans, Inc.<br />
&apos;--- Support: <a class="external-link" href="http://support.softartisans.com"
rel="nofollow">http://support.softartisans.com</a><br />
&apos;--- Sales:   sales@softartisans.com<br />
&apos;-----------------------------------------------------------------</p>

<p>Imports System.Data<br />
Imports System.Data.SqlClient<br />
Imports System.Configuration<br />
Imports SoftArtisans.OfficeWriter.ExcelWriter</p>

<p>Namespace SoftArtisans.OfficeWriter.ExcelWriter.Samples</p>

<p>Public Class TemplateToApp<br />
    Inherits System.Web.UI.Page</p>

<p>        &apos; Both ExcelApplication and ExcelTemplate<br />
        &apos;* objects will be used for this demo.<br />
        &apos;<br />
        Private xlt As ExcelTemplate<br />
        Private xlw As ExcelApplication<br />
        Private wb As Workbook<br />
        Private connString As String =
System.Configuration.ConfigurationManager.AppSettings(&quot;connString&quot;)</p>

<p>    Private Sub Page_Load(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles MyBase.Load<br />
        &apos;Put user code to initialize the page here<br />
    End Sub</p>

<p>    Private Sub btnTemplateToApp_ServerClick(ByVal sender As System.Object, ByVal e
As System.EventArgs) Handles btnTemplateToApp.ServerClick<br />
        GenerateReport()<br />
    End Sub</p>

<p>    &apos;&apos;&apos; &lt;summary&gt;<br />
    &apos;&apos;&apos; Build the report with ExcelApplication<br />
    &apos;&apos;&apos; &lt;/summary&gt;<br />
    Private Sub GenerateReport()<br />
        PopulateTemplate()<br />
```

```
        AddChart()</p>

<p>        &apos; Save the report by streaming it<br />
        &apos;* to the client&apos;s browser<br />
        xlw.Save(wb, Page.Response, &quot;TemplateToApp.xls&quot;, False)<br />
    End Sub</p>

<p>    &apos;&apos;&apos; &lt;summary&gt; Add a column chart to the second worksheet
using ExcelApplication.<br />
    &apos;&apos;&apos; The chart will show data imported with the ExcelTemplate<br />
    &apos;&apos;&apos; object.&lt;/summary&gt;<br />
    Private Sub AddChart()<br />
        &apos; Get the first two worksheets and give them names<br />
        Dim ws As Worksheet = wb.Worksheets(0)<br />
        ws.Name = &quot;Data&quot;</p>

<p>        Dim ws2 As Worksheet =
wb.Worksheets.CreateWorksheet(&quot;ChartSheet&quot;)</p>

<p>        &apos; Create a chart on the second worksheet<br />
        Dim anch As Anchor = ws2.CreateAnchor(0, 0, 50, 50)<br />
        Dim chrt As Chart = ws2.Charts.CreateChart(ChartType.Column.Clustered,
anch)</p>

<p>        &apos; Set series and category data<br />
        Dim srs1 As Series =
chrt.SeriesCollection.CreateSeries(&quot;=Data!B2:B7&quot;)<br />
        chrt.SeriesCollection.CategoryData = &quot;=Data!A2:A7&quot;<br />
        srs1.NameFormula = &quot;=Data!A1&quot;</p>

<p>        &apos; Configure the chart&apos;s legend<br />
        Dim lgnd As Legend = chrt.Legend<br />
        lgnd.Visible = True<br />
        lgnd.Location = Legend.LegendLocation.Right</p>

<p>        &apos; Set the chart&apos;s Title string and display properties.<br />
        chrt.Title.Text = &quot;AdventureWorks Global Sales&quot;<br />
    End Sub</p>

<p>    &apos;&apos;&apos; &lt;summary&gt; Populate the template workbook with
database<br />
    &apos;&apos;&apos; data using the ExcelTemplate object.  The ExcelTemplate<br />
    &apos;&apos;&apos; save() method returns a Workbook.<br />
    &apos;&apos;&apos; /summary&gt;<br />
    Private Sub PopulateTemplate()<br />
        &apos; Create an instance of ExcelTemplate and open<br />
        &apos;* the template workbook<br />
        &apos;<br />
        xlt = New ExcelTemplate</p>

<p>        &apos; Open the template workbook<br />
        Dim templatePath As String =
Page.MapPath(&quot;templates/TemplateToAppTemplate.xls&quot;)<br />
        xlt.Open(templatePath)</p>

<p>        &apos; Get the data from the database and set it as<br />
        &apos;* a data source.<br />
        &apos;<br />
        Dim dt As DataTable = GetData()<br />
```

```
        xlt.BindData(dt,&quot;Sales&quot;,xlt.CreateDataBindingProperties())<br />
        xlt.Process()</p>

<p>        &apos; Here the populated ExcelTemplate object<br />
        &apos;* is being opened as an ExcelApplication Workbook.<br />
        &apos;* The object can now be programatically manipulated<br />
        &apos;* with the ExcelApplication API<br />
        &apos;<br />
        xlw = New ExcelApplication<br />
        wb = xlw.Open(xlt)<br />
    End Sub</p>

<p>    &apos;&apos;&apos; &lt;summary&gt; Query the database for the data that will<br
/>
    &apos;&apos;&apos; be imported by the ExcelTemplate object.<br />
    &apos;&apos;&apos; &lt;/summary&gt;<br />
    &apos;&apos;&apos; &lt;returns&gt;DataTable of data for the
report&lt;/returns&gt;<br />
    Private Function GetData() As DataTable<br />
            Dim sql As String = &quot;SELECT Person.StateProvince.CountryRegionCode As
Country, &quot; + _<br />
                &quot;SUM(Sales.SalesOrderHeader.TotalDue) As TotalSales &quot; + _<br
/>
                &quot;FROM Sales.SalesOrderHeader &quot; + _<br />
                &quot;JOIN Sales.CustomerAddress &quot; + _<br />
                &quot;ON
Sales.SalesOrderHeader.CustomerID=Sales.CustomerAddress.CustomerID &quot; + _<br />
                &quot;JOIN Person.Address &quot; + _<br />
                &quot;ON Sales.CustomerAddress.AddressID=Person.Address.AddressID
&quot; + _<br />
                &quot;JOIN Person.StateProvince &quot; + _<br />
                &quot;ON
Person.StateProvince.StateProvinceID=Person.Address.StateProvinceID &quot; + _<br />
                &quot;GROUP BY Person.StateProvince.CountryRegionCode&quot;</p>

<p>        Dim dt As DataTable = New DataTable<br />
        Dim conn As SqlConnection = New SqlConnection(connString)<br />
        Try<br />
            Dim adpt As New SqlDataAdapter(sql, conn)<br />
            adpt.Fill(dt)<br />
        Finally<br />
            If Not conn Is Nothing Then<br />
                conn.Dispose()<br />
            End If<br />
        End Try</p>

<p>        Return dt<br />
    End Function</p>


<p>#Region &quot; Web Form Designer Generated Code &quot;</p>

<p>    &apos;This call is required by the Web Form Designer.<br />
    &lt;System.Diagnostics.DebuggerStepThrough()&gt; Private Sub
InitializeComponent()</p>

<p>    End Sub<br />
    Protected WithEvents Form1 As System.Web.UI.HtmlControls.HtmlForm<br />
    Protected WithEvents btnTemplateToApp As
```

```
System.Web.UI.HtmlControls.HtmlInputButton</p>

<p>    &apos;NOTE: The following placeholder declaration is required by the Web Form
Designer.<br />
    &apos;Do not delete or move it.<br />
    Private designerPlaceholderDeclaration As System.Object</p>

<p>    Private Sub Page_Init(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles MyBase.Init<br />
        &apos;CODEGEN: This method call is required by the Web Form Designer<br />
        &apos;Do not modify it using the code editor.<br />
        InitializeComponent()<br />
    End Sub</p>

<p>#End Region</p>
```

```
<p>End Class<br />
End Namespace</p>
```