Passing ExcelApplication to ExcelTemplate

Passing ExcelApplication to ExcelTemplate

ExcelWriter allows you to generate a spreadsheet from script alone - using the ExcelApplication object - or from a template spreadsheet and a script, using ExcelTemplate. You can also use ExcelApplication and ExcelTemplate together. This section shows you how to generate a template spreadsheet with ExcelApplication and pass the spreadsheet to ExcelTemplate which then populates the template's data markers.

1. Use ExcelApplication to create a worksheet and enter data marker strings in the worksheet's cells:

```
ExcelApplication xla = new ExcelApplication();
Workbook wb = xla.Create();
Worksheet ws = wb.Worksheets[0];

for (int i = 0; i < colCount; i++)
{
     //--- Form the current data marker.
     //--- Data markers are in this format:
     //--- %%=[dsrcName].[colName]
     string curMarker = String.Format("%%={0}.{1}", dsrcName,
          selectedCols[i].ToString());
     ws[firstRow, firstCol + i].Value = curMarker;
}
```

   A data marker is a cell value beginning with %%= or %%=$ that specifies a variable, an array, or a database column, to insert in the spreadsheet column containing the marker. The data marker may be followed by a modifier. For example, to include column headers in the populated data marker column, the data marker should include the (fieldname) modifier:

```
%%=DataSource.EmployeeID(fieldname)
```

   For more information, see Creating Data Markers.

2. To open the workbook you created as an ExcelTemplate, pass your ExcelApplication and Workbook objects to ExcelTemplate's Open method:

```
ExcelTemplate xlt = new ExcelTemplate();
xlt.Open(xla, wb);
```

3. Connect to a database and execute a query to return a DataTable, DataView, or DataReader to use a data source for template data markers, for example:

```
private DataTable GetEmployeeDataTable()
{
     string EmployeeSQL = "SELECT TOP 10 FirstName + ' ' + LastName As Name, " +
          "Title FROM Employee";

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

     return dt;
}
```

Alternatively, the data source for the template may be a rectangular array.

4. Set the returned DataTable, DataView, or DataReader as the data source for template data markers, and call Process to populate the data markers with data source values:

```
xlt.SetDataSource(dt, dsrcName);
xlt.Process();
```

5. Call Save to save the template on the server, return it in memory, or - as in the following example - stream it to the browser:

```
xlt.Save(Page.Response, "Report.xls", false);
```

Example

```
using SoftArtisans.OfficeWriter.ExcelWriter;

class AppToTemplate : System.Web.UI.Page
{
     protected void Page_Load(object sender, System.EventArgs e)
     {
          //--- Create ExcelApplication, a Workbook, and a Worksheet
          ExcelApplication xla = new ExcelApplication();
          Workbook wb = xla.Create();
          Worksheet ws = wb.Worksheets[0];

          //--- Write ExcelTemplate data markers into the workbook
          ws.Cells[0, 0].Value = "%%=DSrcName.#1(fieldname)";
          ws.Cells[0, 0].Style.Font.Bold = true;
          ws.Cells[1, 0].Value = "%%=DSrcName.#1";

          ws.Cells[0, 1].Value = "%%=DSrcName.#2(fieldname)";
          ws.Cells[0, 1].Style.Font.Bold = true;
          ws.Cells[1, 1].Value = "%%=DSrcName.#2";

          //--- Open the Workbook as ExcelTemplate
          ExcelTemplate xlt = new ExcelTemplate();
          xlt.Open(xla, wb);

          //--- Bind data using ExcelTemplate SetDataSource
          string[,] values = {{"New York", "NY"},
               {"Miami", "FL"},
               {"Boston", "MA"}};
          string[] names = {"City", "State"};
          xlt.SetDataSource(values, names, "DSrcName");
          xlt.Process();
          xlt.Save(Page.Response, "AppToTemplate.xls", false);
     }
}
```

Code Sample: Passing ExcelApplication to ExcelTemplate

[C#] | [VB.NET]