Part 1 - Getting Started

Table of Contents

Getting Started

About templates and merge fields

A WordWriter template is a Microsoft Word file that contains merge fields. A merge field displays a data source field (for example, a database column name). A merge field is created in Microsoft Word and bound in code to a data source. The data source may be an array, a DataSet, a DataTable, or a DataReader. When you run the code, WordWriter populates the merge fields with data source values. A template may contain multiple sets of merge fields. Each set of fields binds to a single data source.

Merge field syntax

- The general syntax is: DataSource.ColumnName
- Ordinal syntax (i.e. #1.#2 for source one, column two) can also be used
- Data source and column names must not include Unicode characters.
- Data source and column names must begin with a letter (A-Z, a-z).
- Data source and column names may include the following characters only:
  ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz1234567890_

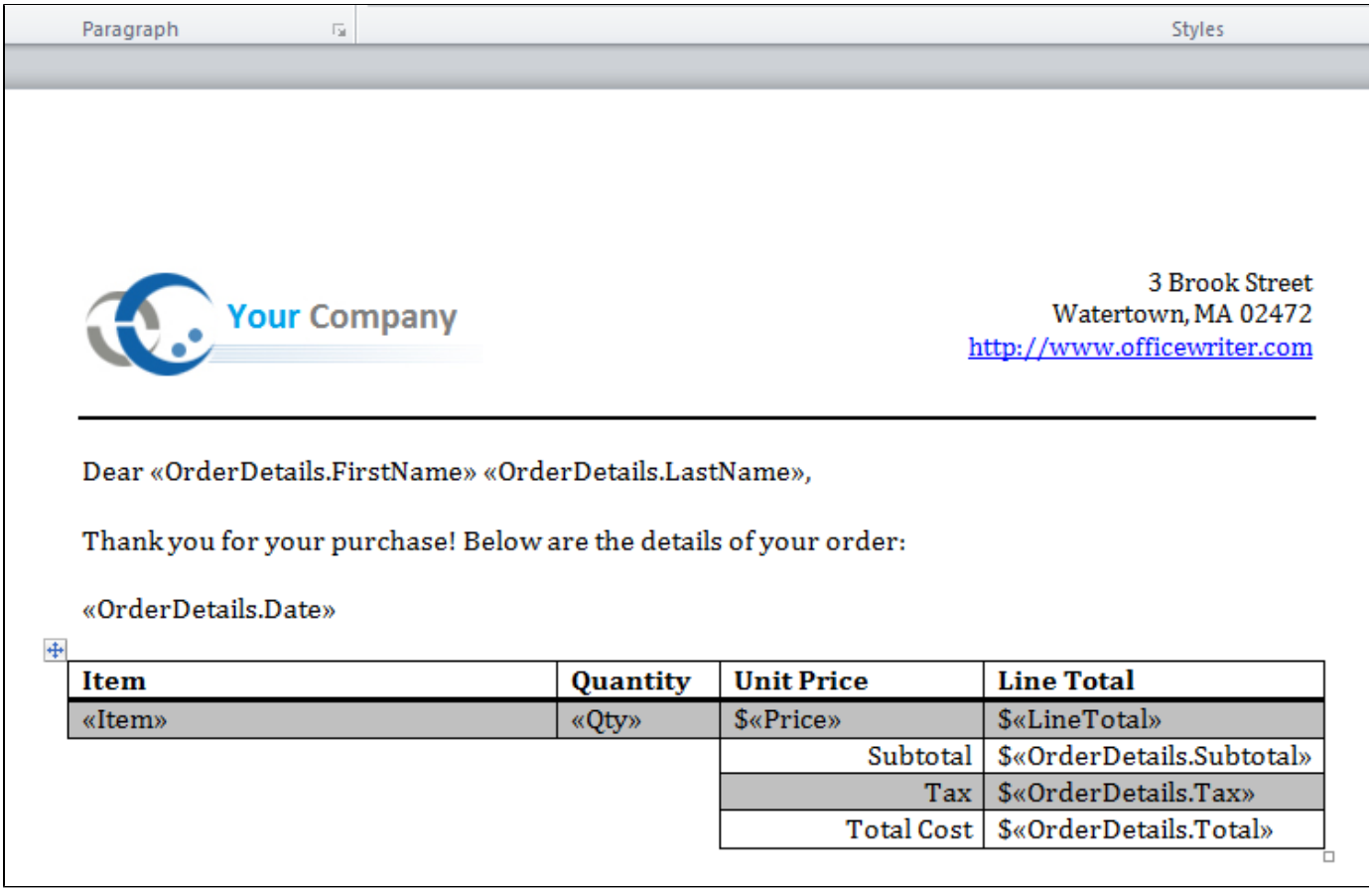
  ⚠ If you assigned a different data source separator, you can also use a "." in your data source and column names.
- Spaces are not allowed anywhere in a merge field.

ⓘ **Use brackets if your column name does not conform to these rules**
The exception of these rules is when a column name is enclosed in brackets (for example, «Products.[Product Category]» . To include spaces or unicode characters in the data source column name, use this format:«DataSource.[Column Name]»

Setting up the Template

Adding merge fields

The final template will resemble this:



Following the Sample Code
In the downloadable WordWriter_Basic_Tutorials.zip under *SalesInvoice*, there is a completed template file located in *SalesInvoice/templates/Part1_Invoice_Template.docx*.

1. Start with a blank .docx file.

2. Add the merge fields for the header. These values are a single row of data called "OrderHeader." The values are "FirstName," "LastName," and "Date."

3. Create a table for the order data and add the merge fields These values are a single row of data called "OrderDetails." In this sample, the values are "Item," "Qty," "Price," "LineTotal," "SubTotal," "Tax," and "Total."

4. Format the currency fields with a formatting switch, such as: "#/ $#,###0.00"

The template should resemble this:

5. The template can be styled as desired. Any formatting applied to a merge field will be persisted when WordWriter populates the data.

Adding a WordWriter Reference in Visual Studio

> **Following the Sample Code**
> In the sample code, the reference to *SoftArtisans.OfficeWriter.WordWriter.dll* has already been added to the *SalesInvoice* project.

Create a .NET project and add a reference to the WordWriter library.

1. Open Visual Studio and create a .NET project.
    - The sample code uses a web application.
2. Add a reference to SoftArtisans.OfficeWriter.WordWriter.dll
    - SoftArtisans.OfficeWriter.WordWriter.dll is located under **Program Files > SoftArtisans > OfficeWriter > dotnet > bin**

Writing the Code

> **Following the Sample Code**
> There is a sample web application page `Part1.aspx` and code behind `Part1.aspx.cs` available in the **SalesInvoice/** directory that shows the completed code.

1. Include the SoftArtisans.OfficeWriter.WordWriter namespace in the code behind.

```
using SoftArtisans.OfficeWriter.WordWriter;
```

2. In the method that will actually run the report, instantiate the `WordTemplate` object.

```
WordTemplate WT = new WordTemplate();
```

3. Open the template file with the `WordTemplate.Open` method.

```
WT.Open(Page.MapPath("//templates//Part1_Invoice_Template.docx"));
```

4. Create an `object` array for the header values and a `string` array for the column names.

`WordTemplate` can be bound to numerous types of .NET data structures: single variables, arrays (1-D, jagged, multi-dimensional), `DataSet`,`DataTable`, `IDataReader` etc. The source of the data can come from anywhere.

Some of the aforementioned structures have built in column names, such as the `DataTable`. When working with arrays, which don't have built in column names, you have to define the column names in a separate `string` array.

```
object[] orderHeader = { "Jane", "Doe", DateTime.Now.ToString("MM/dd/yy")};
string[] orderHeaderColNames = { "FirstName", "LastName", "Date"};
```

5. Create two arrays as the OrderDetail data set.

```
object[] detailsArray = {"Sport-100 Helmet, Black", 3, 34.99, 104.97, 104.97, 4.46,
109.43 };

string[] detailColNames = {"Item", "Qty", "Price", "LineTotal", "Subtotal", "Tax",
"Total"  };
```

6. Use the `WordTemplate.SetDataSource` method to bind the order info to the merge fields in the template file.

`SetDataSource()` binds a single row of data to the template.

```
WT.SetDataSource(orderHeader, orderHeaderColNames, "OrderHeader");
```

7. Use `SetDataSource()` to bind the order details arrays. Note that the data source name is the last string.

```
WT.SetDataSource(detailsArray, detailColNames, "OrderDetails");
```

8. Call `WordTemplate.Process` to import the data into the file.

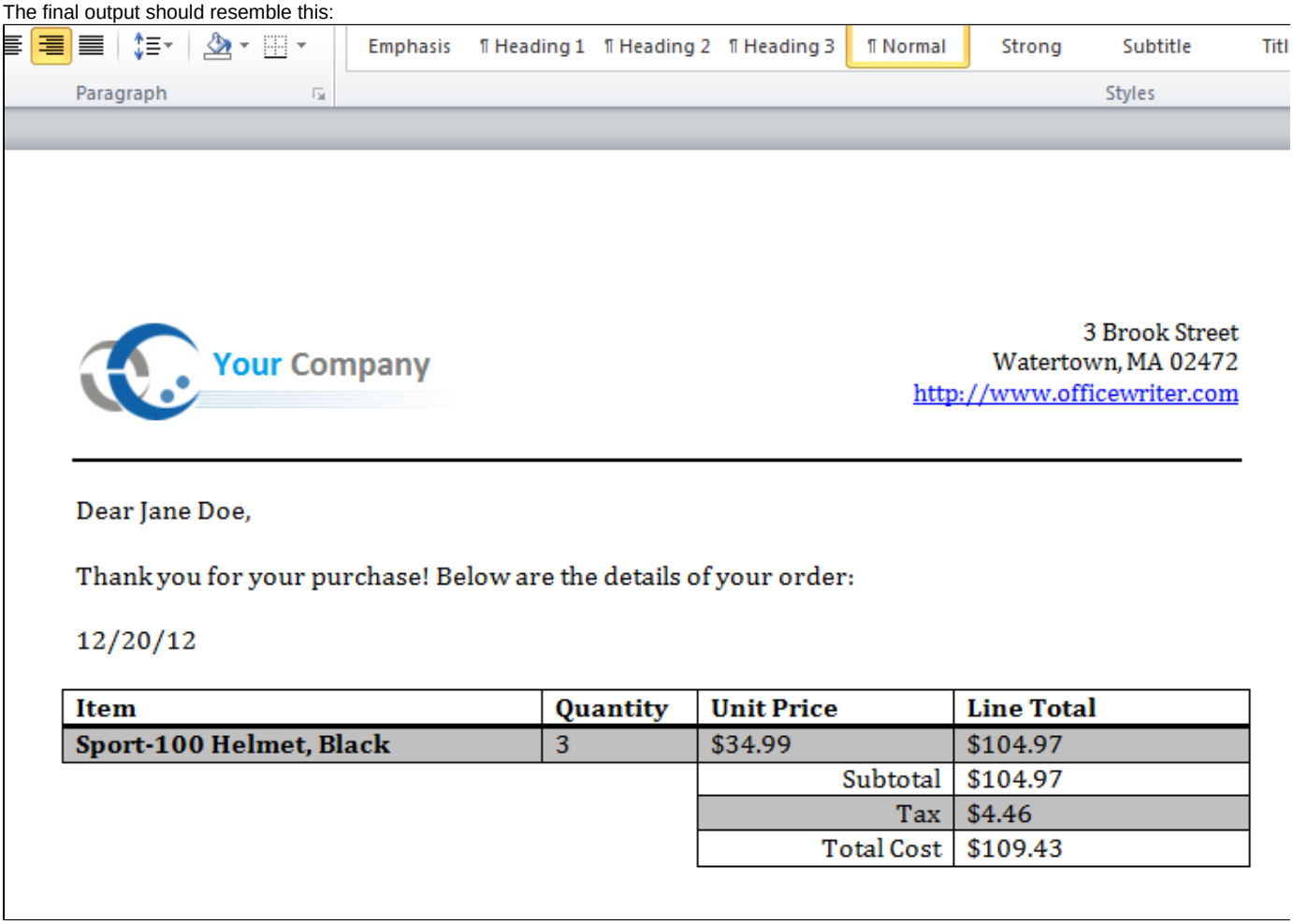
```
WT.Process();
```

9. Call `WordTemplate.Save` to save the output file.

`WordTemplate` has several output options: save to disk, save to a stream, stream the output file in a page's `Response` inline or as an attachment.

```
WT.Save(Page.Response, "Part1_Output.docx", false);
```

The final output should resemble this:



Final Code

```
using SoftArtisans.OfficeWriter.WordWriter;
...

//Instantiate a new WordTemplate object
WordTemplate WT = new WordTemplate();

//Open the template file
WT.Open(Page.MapPath("//templates//Part1_Invoice_Template.docx"));

//Create the array of header values
object[] orderHeader = { "Jane", "Doe", DateTime.Now.ToString("MM/dd/yy")};
//Create the array of column names
string[] orderHeaderColNames = { "FirstName", "LastName", "Date"};

//Create the array of details values
object[] detailsArray = {"Sport-100 Helmet, Black", 3, 34.99, 104.97, 104.97, 4.46,
109.43 };
//Create the array of column names
string[] detailColNames = {"Item", "Qty", "Price", "LineTotal", "Subtotal", "Tax",
"Total"  };

//Set the data sources to import a single row of data for each source
WT.SetDataSource(orderHeader, orderHeaderColNames, "OrderHeader");
WT.SetDataSource(detailsArray, detailColNames, "OrderDetails");

//Process to import the data to the template
WT.Process();

WT.Save(Page.Response, "Part1_Output.docx", false);
```

Downloads

You can download the code for the Basic WordWriter Tutorials as a Visual Studio solution, which includes the Simple Expense Summary.

- WordWriter_Basic_Tutorials.zip

Next Steps

Continue on to Part 2: Repeat Blocks