

# Part 2 - Repeat Blocks

## Table of Contents

- [Intro](#)
- [Changing the Template](#)
- [Writing the Code](#)
- [Final Code](#)
- [Downloads](#)

## Intro



This is Part 2 of a 2-part tutorial series for the [Sales Invoice](#) scenario. It is recommended that you complete [Part 1 - Getting Started](#) before starting this section.



### Following the Sample

There is a downloadable [WordWriter\\_Basic\\_Tutorials.zip](#) with completed templates and code. The completed example of the template is available under *templates/Part2\_Invoice\_Template.xlsx*. The code for this part of the tutorial can be found in *Part2.aspx.cs*.

This part focuses on adding repeating data to an order summary. There are slight modifications to the template and code from Part 1.

## Changing the Template

The starting template is from [Part 1 - Getting Started](#):

| Paragraph                                                                                                                                                                                                                                                                                                                                                                                        | Styles |
|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|--------|
| <div><div>3 Brook Street<br/>Watertown, MA 02472<br/><a href="http://www.officewriter.com">http://www.officewriter.com</a></div></div>                                                                                                                                                                        |        |
| Dear «OrderDetails.FirstName» «OrderDetails.LastName»,                                                                                                                                                                                                                                                                                                                                           |        |
| Thank you for your purchase! Below are the details of your order:                                                                                                                                                                                                                                                                                                                                |        |
| «OrderDetails.Date»                                                                                                                                                                                                                                                                                                                                                                              |        |
| <div><div><div><div><div>Item</div><div>Quantity</div><div>Unit Price</div><div>Line Total</div></div><div>«Item»</div><div>«Qty»</div><div>\$«Price»</div><div>\$«LineTotal»</div></div><div><div>Subtotal</div><div>\$«OrderDetails.Subtotal»</div></div><div><div>Tax</div><div>\$«OrderDetails.Tax»</div></div><div><div>Total Cost</div><div>\$«OrderDetails.Total»</div></div></div></div> |        |

Repeat blocks are used to [import multiple rows](#). A repeat block is a fragment in the template document - defined by a Word bookmark - that contains [merge fields](#) and that will be repeated for each row in a data source. To import multiple rows from a single data source, create a repeat block in the template and, in the WordWriter code, call `SetRepeatBlock` to bind the repeat block to a data source.

In this sample, the repeat block is added to the order info merge fields. This bookmark is called "Repeat."

Add a bookmark around the data you wish to import.



3 Brook Street  
Watertown, MA 02472  
<http://www.officewriter.com>

Dear «OrderDetails.FirstName» «OrderDetails.LastName»,

Thank you for your purchase! Below are the details of your order:

«OrderDetails.Date»

| Item   | Quantity | Unit Price | Line Total                |
|--------|----------|------------|---------------------------|
| «Item» | «Qty»    | \$«Price»  | \$«LineTotal»             |
| Repeat |          | Subtotal   | \$«OrderDetails.Subtotal» |
|        |          | Tax        | \$«OrderDetails.Tax»      |
|        |          | Total Cost | \$«OrderDetails.Total»    |

## Writing the Code



### Following the Sample Code

There is a sample web application page `Part2.aspx` and code behind `Part2.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//Part2_Invoice_Template.docx"));
```

4. Create an object array for the header and total 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"), 13139.51, 558.43, 13697.94 };
string[] orderHeaderColNames = { "FirstName", "LastName", "Date", "Subtotal", "Tax",
    "Total" };
```



#### Following the Sample

In the sample project, we are parsing CSV files with query results, rather than querying a live database. The CSV files are available under the *data\_directory*. There is a copy of the CSV parser, *GenericParsing.dll* in the *\_bin* directory of the project. *GetCSVData* is defined in *Part1.aspx.cs* in a region marked *Utility Methods*.

If you are following in your own project and would like to parse the CSV files as well, you will need to:

- Add a reference to *GenericParsing.dll*.
- Include *GeneringParsing* at the top of your code.
- Add the *GetCSVData* method that can be found in the sample code.

5. Get the datatable for the repeat block.

```
DataTable dtOrderInfo = GetCSVData("//data//OrderInfo.csv");
```

6. Use *WordTemplate.SetRepeatBlock* to pass the data and the bookmark name.

```
WT.SetRepeatBlock(dtOrderInfo, "Repeat");
```

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

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

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, "Part2_Output.docx", false);
```

The final output should resemble this:



3 Brook Street  
Watertown, MA 02472  
<http://www.officewriter.com>

Dear Jane Doe,

Thank you for your purchase! Below are the details of your order:

12/20/12

| Item                    | Quantity | Unit Price | Line Total |
|-------------------------|----------|------------|------------|
| Sport-100 Helmet, Black | 3        | \$34.99    | \$104.97   |
| Sport-100 Helmet, Red   | 1        | \$34.99    | \$34.99    |
| Road-650 Red, 44        | 3        | \$782.99   | \$2348.97  |
| LL Road Frame - Red, 48 | 2        | \$337.22   | \$674.44   |
| Road-450 Red, 44        | 2        | \$1457.99  | \$2915.98  |
| Road-650 Red, 60        | 3        | \$782.99   | \$2348.97  |
| LL Road Frame - Red, 62 | 1        | \$337.22   | \$337.22   |
| Road-450 Red, 58        | 3        | \$1457.99  | \$4373.97  |
| Subtotal                |          |            | \$13139.51 |
| Tax                     |          |            | \$558.43   |
| Total Cost              |          |            | \$13697.94 |

**Final Code**

```

using SoftArtisans.OfficeWriter.WordWriter;
...
//Instantiate a new WordTemplate object
WordTemplate WT = new WordTemplate();

//Open the template file
WT.Open(Page.MapPath("//templates//Part2_Invoice_Template.docx"));

//Create the array of header values
object[] detailsArray
    = { "Jane", "Doe", DateTime.Now.ToString("MM/dd/yy"), "13139.51", "558.43",
    "13697.94" };
//Create the array of column names
string[] detailColNames = { "FirstName", "LastName", "Date", "Subtotal", "Tax",
    "Total" };

//Get the order info datatable using GenericParser
DataTable dtOrderInfo = GetCSVData("//data//OrderInfo.csv");

//Set the repeat block to bind the data for multiple order items
WT.SetRepeatBlock(dtOrderInfo, "Repeat");
//Set the header data source to import a single row of data
WT.SetDataSource(orderHeader, orderHeaderColNames, "OrderHeader");

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

WT.Save(Response, "Part2_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](#)