

WordWriter Formulas

Intro

This demo illustrates some of the formulas available for WordTemplate in OfficeWriter 4.1 and above.

While Microsoft Word provides a limited number of formulas, they are restrictive. Specifically, Word formulas must be contained in a table, and must take either ABOVE, LEFT, or a list of values.

With OfficeWriter 4.1 and above, WordTemplate provides a selection of formulas that enhance Word's built-in formulas. A WordTemplate formula is specified as a mergefield and has a special syntax: «=SUM(datasource.fieldname)». A WordTemplate formula operates on a column of values in the data source and can be placed anywhere in a document. In OfficeWriter 4.1 and above, the following WordTemplate formulas are available:

- AVERAGE
- COUNT
- COUNTA
- MAX
- MIN
- PRODUCT
- STDEV
- STDEVP
- SUM
- VAR
- VARP

Code

```
public void GenerateDocument()
{
    WordTemplate wt = new WordTemplate();

    // Create the array of mergefield names
    string[] fields = new string[] { "TestDate", "School", "Town", "Version"
};

    // Create the array of values, each of which corresponds to a mergefield
above
    object[] values = new object[] {
        new DateTime(2009, 11, 16).ToShortDateString(), "Samuel Adams HS",
        "Milbury", wt.Version};

    // Open the template document
    wt.Open(@"..\..\WordTemplateFiles\TestScoreReportTemplate.docx");

    // Set the main document data source
    wt.SetDataSource(values, fields, "");

    // Retrieve test score data
    DataTable dt = GetScores();

    // In a repeat block, a new entry is inserted for each row in the
    // data source. When the data source is a DataTable, the mergefield
    // names are the same as the column names; so there is no need to
    // specify mergefield names as we did above. However, we do have to set
    // a bookmark, "Items", so that wordwriter knows where to repeatedly bind
the data.
```

```

        wt.SetRepeatBlock(dt, "Items");

        // Populate the template
        wt.Process();

        // Save the document to the disc in the desired location
        wt.Save(@"..\..\WordOutputFiles\ScoreReport_output.docx");
    }

private DataTable GetScores()
{
    DataTable dt = new DataTable();

    dt.Columns.Add("StudentID", typeof(string));
    dt.Columns.Add("ReadingScore", typeof(int));
    dt.Columns.Add("WritingScore", typeof(int));
    dt.Columns.Add("MathScore", typeof(int));
    dt.Columns.Add("TotalScore", typeof(int));

    Random rand = new Random();

    int rScore, wScore, mScore;
    for (int i = 0; i < 30; ++i)
    {
        rScore = rand.Next(400, 800);
        wScore = rand.Next(400, 800);
        mScore = rand.Next(400, 800);
        dt.Rows.Add(
            string.Format("S{0:000000}", rand.NextDouble() * 1000000),
            rScore, wScore, mScore, rScore + wScore + mScore
        );
    }

    DataTable dtSorted = dt.Clone();
    foreach (DataRow row in dt.Select("", "StudentID asc"))
        dtSorted.ImportRow(row);

    return dtSorted;
}

```



Downloads

Template: TestScoreReportTemplate.docx
Output: ScoreReport_output.docx