

Sample Expense Report

Intro

This sample uses ExcelTemplate to populate a generic expense report template.

This example demonstrates how to use single object data markers (ones that start with `%%=$`) to import the data for the header by calling the `BindCellData` method multiple times. It also demonstrates how to use 2-D array data markers to import expense data from an Object array by calling the `BindData` method once.

Requirements

This sample requires OfficeWriter Enterprise Edition to be installed because the OfficeWriter Grouping and Nesting is only available in the Enterprise Edition of the product.

Code

[illegible]

```

        // Create an instance of ExcelTemplate
        ExcelTemplate xlt = new ExcelTemplate();

        // Open the template workbook
        xlt.Open(@"..\..\ExcelTemplateFiles\ExpenseReportTemplate.xlsx");

        // Pass the strings to the BindCellData method
        DataBindingProperties cellBindingProperties =
xlt.CreateDataBindingProperties();
        xlt.BindCellData(empPurpose, "EmpPurpose", cellBindingProperties);
        xlt.BindCellData(empName, "EmpName", cellBindingProperties);
        xlt.BindCellData(empDept, "EmpDept", cellBindingProperties);
        xlt.BindCellData(empSSN, "EmpSSN", cellBindingProperties);
        xlt.BindCellData(empPosition, "EmpPosition", cellBindingProperties);
        xlt.BindCellData(empID, "EmpID", cellBindingProperties);
        xlt.BindCellData(empManager, "EmpManager", cellBindingProperties);
        xlt.BindCellData(dateFrom, "DateFrom", cellBindingProperties);
        xlt.BindCellData(dateTo, "DateTo", cellBindingProperties);

        // Limit the number of rows to the maximum number of rows that Excel
        //can support. ExcelTemplate expects the array's first index to be
        //the row number and the second index to be the column number. Our
        //array is the opposite, so we have to transpose the data.

        DataBindingProperties bindingProperties =
xlt.CreateDataBindingProperties();
        bindingProperties.MaxRows = ExcelTemplate.ALL_ROWS;
        bindingProperties.Transpose = true;
        xlt.BindData(values, colNames, "Exp", bindingProperties);

        // Call the Process() method to populate the template
        //with the data source values

        xlt.Process();

        // Save the report
        xlt.Save(@"..\..\ExcelOutputFiles\ExpenseReport_output.xlsx");
    }

}

```

--

Downloads

- Template: [ExpenseReportTemplate.xlsx](#)
- Output: [ExpenseReport_output.xlsx](#)