

# ExcelTemplate.BindData(System.Data.DataTable, String, DataBindingProperties)

## Description

Sets an ADO.NET DataTable as a data source to bind to template data markers.

**C#**

```
public virtual void BindData(System.Data.DataTable source, System.String
dataSourceName, DataBindingProperties property)
```

**vb.net**

```
Public Overridable Sub BindData(ByVal source As System.Data.DataTable, ByVal
dataSourceName As String, ByVal [property] As DataBindingProperties)
```

## Parameters

### **source**

The DataTable to use as the data source.

### **dataSourceName**

The name of the set of data markers at which to insert the values imported from the data source. dataSourceName must be specified, but can be left as null or an empty string if this is the first data source bound AND the data markers in the template use the [short data marker syntax](#) or refer to the datasource by number rather than name. Note: dataSourceName does not include a data marker's column name, for example, the dataSourceName for %=Products.ProductID is "Products."

### **property**

The [DataBindingProperties](#) object which contains information about how the data should be bound to the template.

### **property**

The [DataBindingProperties](#) object which contains information about how the data should be bound to the template. property Must be specified, but the DataBindingProperties need not be set beforehand. To bind data to a template with the default DataBindingProperties, pass in ExcelTemplate.CreateDataBindingProperties() as the property value. Otherwise, use the ExcelTemplate.CreateDataBindingProperties() method to generate a new DataBindingProperties object and set the [DataBindingProperties.MaxRows](#), [DataBindingProperties.Transpose](#), and/or [DataBindingProperties.WorksheetName](#) properties for the workbook.

## Exceptions

### **ArgumentNullException**

BindData will throw this exception if null (C#) or Nothing (VB.NET) is passed to the method.

### **SARuntimeException**

BindData will throw this exception if the data source contains more rows than the worksheet can hold.

If there is more than one data marker referring to a data source and the data source is forward only, the exception will be thrown only if the source is larger than all bindings can hold.

## ArgumentException

## Remarks

You can set several data sources for a single template. Use the following methods to set template data sources: [BindCellData](#), [BindColumnData](#), [BindRowData](#), and [BindData](#).

## Examples

C#

```
ExcelTemplate xlt = new ExcelTemplate();
OleDbConnection Conn = new OleDbConnection();
DataTable EmployeeDt = null;
try
{
    Conn.ConnectionString = Application["connstring"].ToString();

    ///--- SQL Query for employee information
    string EmployeeSQL = "SELECT FirstName + ' ' +
        LastName As Name, Title " +
        "FROM Employees WHERE EmployeeID=?";
    OleDbCommand CmdEmployee = new OleDbCommand(EmployeeSQL, Conn);
    CmdEmployee.Parameters.Add("@EmployeeID", EmployeeId);
    OleDbDataAdapter AdptEmployee = new OleDbDataAdapter(CmdEmployee);
    EmployeeDt = new DataTable();
    AdptEmployee.Fill(EmployeeDt);
}
xlt.BindData(EmployeeDt,
    "Employee",
    xlt.CreateDataBindingProperties());
```

```
Dim xlt As New ExcelTemplate()
Dim Conn As New OleDbConnection()
Dim EmployeeDt As DataTable = Nothing
Try
    Conn.ConnectionString = Application("connstring").ToString()

    '--- SQL Query for employee information
    Dim EmployeeSQL As String = "SELECT FirstName & ' ' & _
        LastName As Name, Title " & _
        "FROM Employees WHERE EmployeeID=?"
    Dim CmdEmployee As New OleDbCommand(EmployeeSQL, Conn)
    CmdEmployee.Parameters.Add("@EmployeeID", EmployeeId)
    Dim AdptEmployee As New OleDbDataAdapter(CmdEmployee)
    EmployeeDt = New DataTable()
    AdptEmployee.Fill(EmployeeDt)
End Try
xlt.BindData(EmployeeDt, _
    "Employee", _
    xlt.CreateDataBindingProperties())
```