

Mailing Labels Demo

Intro

This demo reads customer data from a database and creates standard-sized 5160 mailing labels. Put some label paper into your printer and see for yourself!

The template document has a table defined in it that exactly matches the dimensions of Avery 5160-sized mailing labels. You can take this template document and resize the table to match any size labels.

Code

```
public void GenerateDocument()
{
    // Get the addresses from a datasource and return them as properly
    formatted addresses
    string[] AddressArray = GetCustomerAddressStrings();

    // Create a DataTable with one column for each
    // label across

    DataTable AddressTable = new DataTable();
    AddressTable.Columns.Add("Address1");
    AddressTable.Columns.Add("Address2");
    AddressTable.Columns.Add("Address3");

    string[] Row = new string[AddressTable.Columns.Count];

    int addridx = 0;
    // Set an infinite loop, break is called inside
    while (addridx < AddressArray.Length)
    {
        // Loop once for every Template column
        for (int i = 0; i < Row.Length; i++)
        {
            // If there's a value left in mAddressValues, use it
            // Otherwise, fill an empty string

            if (addridx < AddressArray.Length)
                Row[i] = AddressArray[addridx];
            else
                Row[i] = String.Empty;

            addridx++;
        }

        // Add the data row after all columns are filled
        AddressTable.Rows.Add(Row);
    }

    // Create an instance of WordTemplate
```

```

WordTemplate wt = new WordTemplate();

// Open the template document
string templatePath = @"..\..\WordTemplateFiles\MailLabelTemplate.docx";
wt.Open(templatePath);

// Set the LabelRow repeat block with the AddressTable table
wt.SetRepeatBlock(AddressTable, "LabelRow");

// Process the template to populate the values
wt.Process();

// Save the document

wt.Save(@"..\..\WordOutputFiles\MailLabels_output.docx");
}

//Create a datatable with all then customer information and return it
private DataTable GetCustomersData(){
    DataTable dt = new DataTable();
    dt.Columns.Add("ContactName");
    dt.Columns.Add("Address");
    dt.Columns.Add("City");
    dt.Columns.Add("Region");
    dt.Columns.Add("PostalCode");
    dt.Columns.Add("Country");

    dt.Rows.Add(new Object[] { "Incomparable Bicycle Store", "99 Edgewater
Drive", "Norwood",
    "MA", "2062", "United States" });
    dt.Rows.Add(new Object[] { "Wholesale Bikes", "58 Teed Drive", "Randolph",
    "MA", "2368", "United States" });
    dt.Rows.Add(new Object[] { "Bikes Anyone?", "Ames Plaza", "Saugus",
    "MA", "1906", "United States" });
    dt.Rows.Add(new Object[] { "Purchase Mart", "Wrentham Village",
"Wrentham",
    "MA", "2093", "United States" });
    return dt;
}

// Get the address items from the DataTable, format
// them, and return them in a string array

private string[] GetCustomerAddressStrings()
{
    // Get the DataTables with the customer information
    DataTable dt = GetCustomersData();
    /* this is the format string for an address */
    string FormatTemplate = "{0}\n" +           // Name
        "{1}\n" +           // Street
        "{2}, {3} {4}\n" +   // City, State Zip
        "{5}";              // Country

    ArrayList al = new ArrayList();

    // Copy the DataTable row values into strings
    // and put them in an ArrayList

```

```
foreach (DataRow dr in dt.Rows)
{
    string address = String.Format(FormatTemplate,
        dr["ContactName"], dr["Address"],
        dr["City"], dr["Region"], dr["PostalCode"], dr["Country"]);

    al.Add(address);
}

// Return the values as a string array (string[])
return (string[])al.ToArray(typeof(string));
```

}

Downloads

Template: [MailLabelTemplate.docx](#)

Output: [MailLabels_output.docx](#)