Create a Master-Detail view in Silverlight using ComponentOne DataGrid™ for Silverlight

Article ID: 2029
Applies To: Studio Enterprise
Author: Sheela Nath
Published On: 8/13/2009

Introduction

One of the most important aspects to consider when displaying data is a way to show how all the data interrelates. While a massive grid layout showing one record after another might help visualize all the data, it does nothing in regards to explaining why the data exists or what it can be used for. One way to help organize data is by setting up a Master-Detail relationship. In a Master-Detail relationship all the detail objects displayed are related to the selected master object. In this article we are going to walk through just how easy it is to use ComponentOne DataGrid for Silverlight to add a Master-Detail view to your Silverlight application.

About ComponentOne DataGrid for Silverlight

ComponentOne DataGrid for Silverlight is a robust data-bound Silverlight DataGrid control that makes it easy to display, edit, and analyze tabular data in Silverlight applications. DataGrid for Silverlight is part of our Silverlight control suite, ComponentOne Studio for Silverlight.

Prerequisite

For this tutorial, you must have Microsoft Visual Studio 2008 SP1, the Silverlight Tools for Visual Studio 2008 SP1, and ComponentOne Studio for Silverlight installed. A free 30-day trial and yearly subscriptions of ComponentOne Studio for Silverlight are available from the ComponentOne website.

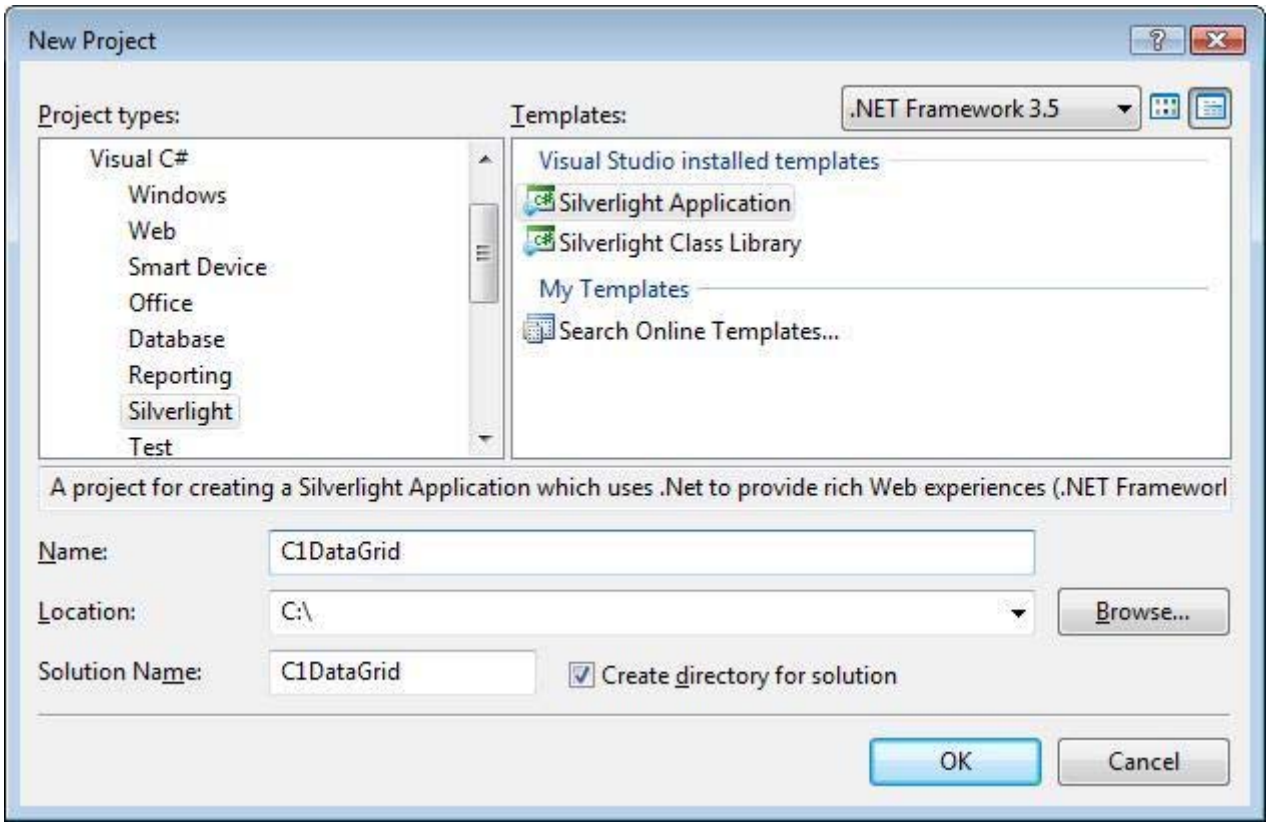
Download Free Trial

The Silverlight Tools for Visual Studio 2008 SP1 can be downloaded from Microsoft's Silverlight site.

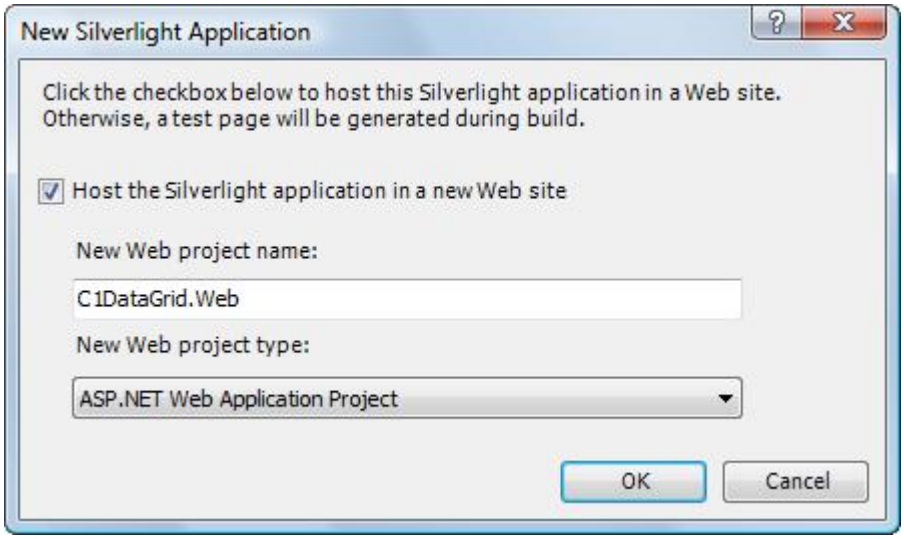
Download Silverlight Tools for Visual Studio 2008 SP1

Step 1: Create a Silverlight Application

Open Microsoft Visual Studio 2008, and select "File | New Project" from the main menu. Under "Visual C#" select "Silverlight" and in the Templates box select "Silverlight Application". Name the application C1DataGrid and then click **OK**.

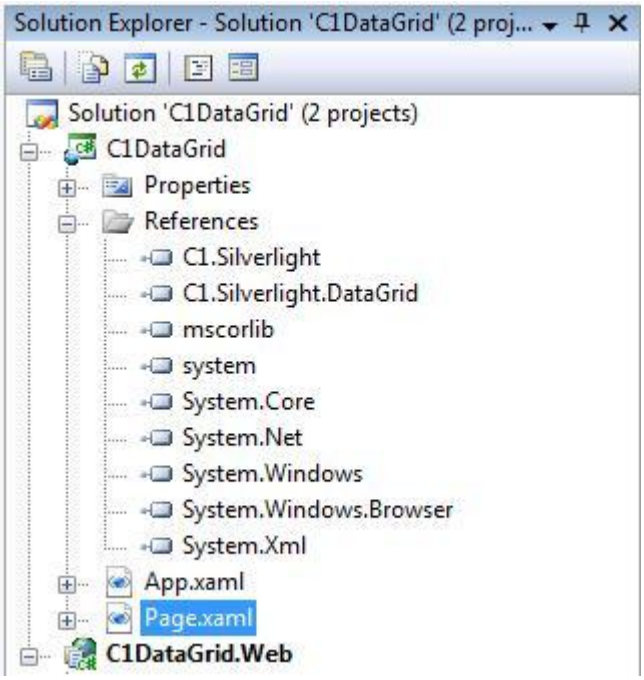


After you click **OK**, you will see another box inquiring as to how you want to host the Silverlight application. For this tutorial, select “Host the Silverlight application in a new Web site” and next to New Web Project Type make sure “ASP.NET Web Application Project” is selected. Then click **OK**.



Step 2: Add ComponentOne DataGrid for Silverlight to the Project

First you'll need to add references to C1.Silverlight.dll and C1.Silverlight.DataGrid.dll. To do so, right-click on "References" in the Solution Explorer, choose "Add Reference", and browse to the needed DLLs. After adding them, the Solution Explorer should look like the following image:



Next, add a reference to the grid in the XAML markup. Below xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml" add the following:

```
xmlns:c1grid="clr-namespace:C1.Silverlight.DataGrid;assembly=C1.Silverlight.DataGrid"
```

Now add the grid to the XAML markup. To do so, click in between the <Grid x:Name="LayoutRoot"></Grid> tags and type the following:

```
<c1grid:C1DataGrid></c1grid:C1DataGrid>
```

This adds the C1DataGrid control to the project.

Name the grid so it may be referenced in the code. Add a name property to the grid tag so it looks like the following:

```
<c1grid:C1DataGrid x:Name="c1datagrid"></c1grid:C1DataGrid>
```

The XAML markup should now look like the following:

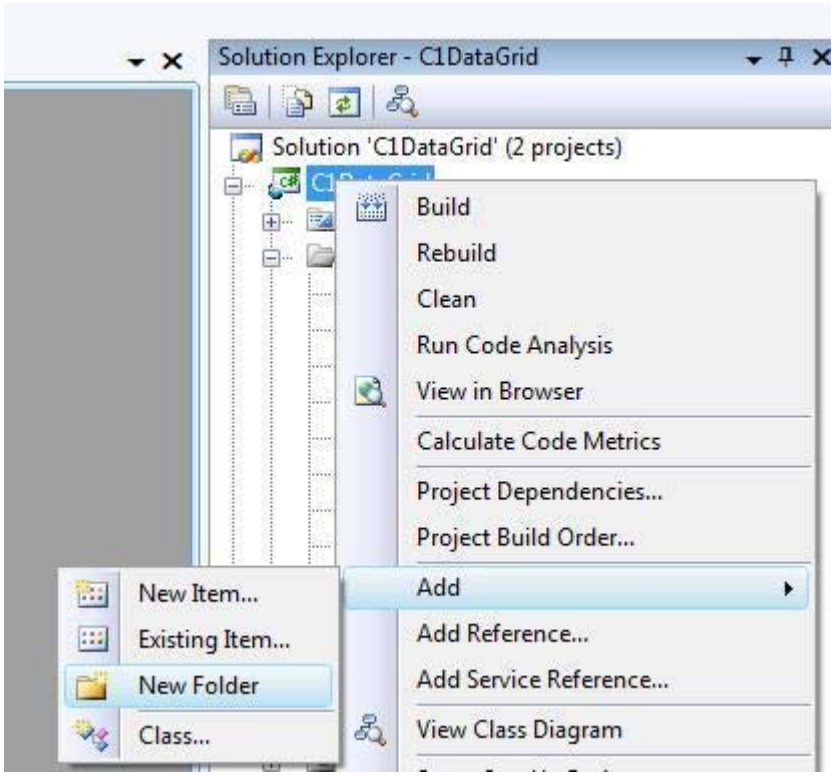
```
<UserControl x:Class="C1DataGrid.MainPage"
    xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
    xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
    xmlns:c1grid="clr-namespace:C1.Silverlight.DataGrid;assembly=C1.Silverlight.DataGrid"
xmlns:d="http://schemas.microsoft.com/expression/blend/2008" xmlns:mc="http://schemas.openxmlformat
    mc:Ignorable="d" d:DesignWidth="640" d:DesignHeight="480">
    <Grid x:Name="LayoutRoot">
        <c1grid:C1DataGrid x:Name="c1datagrid">

        </c1grid:C1DataGrid>
    </Grid>
</UserControl>
```

Step 3: Add a Data Source to the Project

In this step you'll add a data source to your application and add external files to set up the data source. Note that to simplify the tutorial, this step uses files included with the C1DataGrid_Demo samples included with the Studio for Silverlight installation.

First, In the Solution Explorer window, right-click the C1DataGrid project and select "Add | New Folder". Rename the folder "Resources".

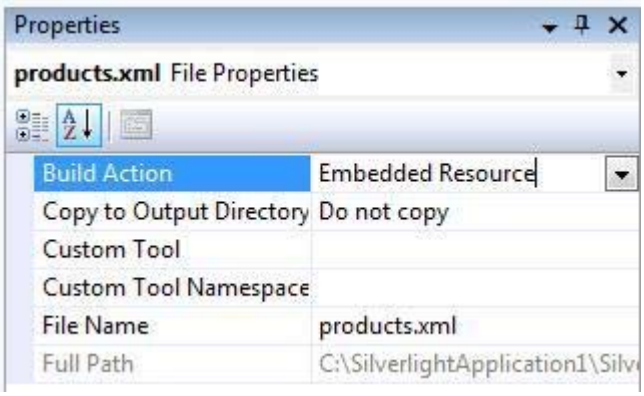


Next, right-click on the Resources folder and select "Add | Existing Item".

In the **Add Existing Item** dialog box, navigate to the C1DataGrid_Demo sample folder, locate and select the products.xml file, and click **Add**. This file provides that data you'll use in the project.

Note that, by default, products.xml will be installed in the* **C:\Program Files\ComponentOne\Studio for Silverlight\Samples\C1.Silverlight.DataGrid\C1DataGrid_Demo\C1DataGrid_Demo\Resources *directory.*

Select the "products.xml" file in the Solution Explorer, and in the Properties window set the **Build Action** property to "Embedded Resource".



In the Solution Explorer window, right-click the C1DataGrid project and select "Add | Existing" Item.

In the **Add Existing Item** dialog box, navigate to the C1DataGrid_Demo sample folder, select the "Data.cs" file, and click **Add**. This file contains code to set up the data source.

Note that, by default, Data.cs will be installed in the* **C:\Program Files\ComponentOne\Studio for Silverlight\Samples\C1.Silverlight.DataGrid\C1DataGrid_Demo\C1DataGrid_Demo *directory.*

Step 4: Add a Data Source to the Project

In this step you'll finish setting up the row details section of the grid. You'll add a RowDetailsTemplate to set the appearance of the details row, and you'll add code to set up the details row behavior.

Add the following <c1grid:C1DataGrid.RowDetailsTemplate></c1grid:C1DataGrid.RowDetailsTemplate> tags between the <c1grid:C1DataGrid></c1grid:C1DataGrid> tags so that it appears similar to the following:

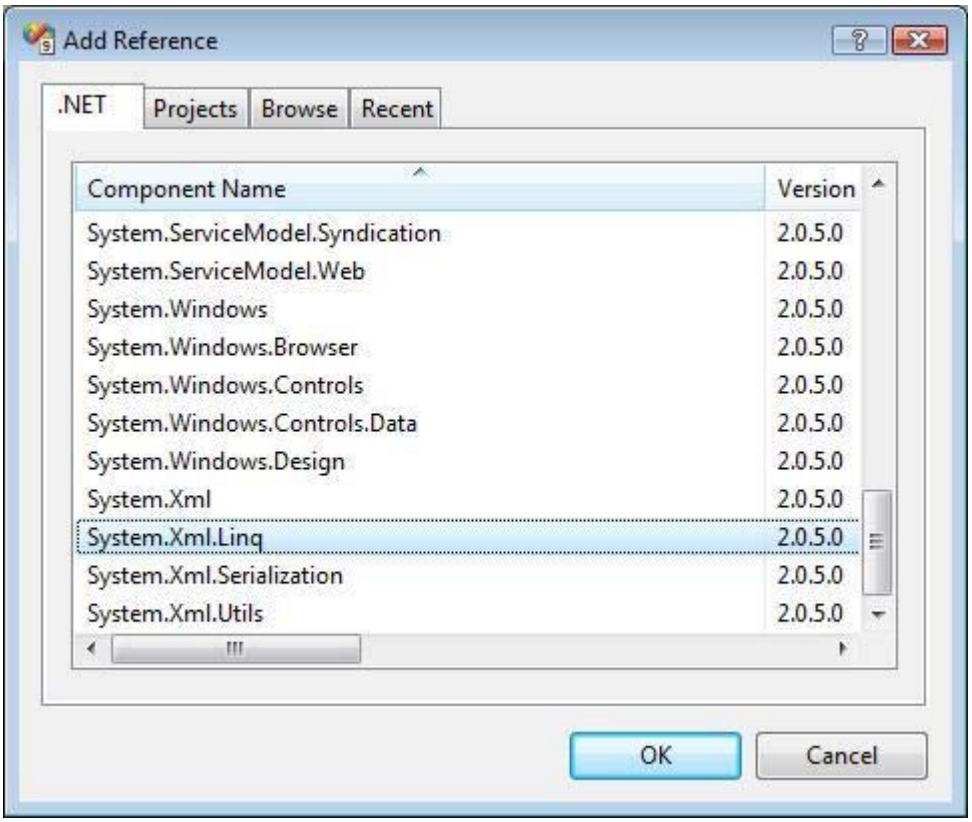
```
<c1grid:C1DataGrid x:Name="c1datagrid">
          <c1grid:C1DataGrid.RowDetailsTemplate>
              <DataTemplate>
                  <c1grid:C1DataGrid HeadersVisibility="Column"  Margin="5"/>
              </DataTemplate>
          </c1grid:C1DataGrid.RowDetailsTemplate>
      </c1grid:C1DataGrid>
```

Add markup in the <c1grid:C1DataGrid> tag to that it appears similar to the following:

```
<c1grid:C1DataGrid x:Name="c1datagrid" CanUserAddRows="False" Margin="5" LoadedRowDetailsPresenter='
```

This will customize the appearance of the grid and add event handlers.

Next, right-click the project and select "Add Reference". In the **Add Reference** dialog box, locate System.Xml.Linq and click **OK** to add the reference.



Now that you've added the reference, it's time to switch over to the code and add some imports statements. Add import statements so they look like the following code:

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Net;
using System.Windows;
using System.Windows.Controls;
using System.Windows.Documents;
using System.Windows.Input;
using System.Windows.Media;
using System.Windows.Media.Animation;
using System.Windows.Shapes;
using System.Xml.Linq;
using C1.Silverlight.DataGrid;
using C1DataGrid_Demo;
```

Add code to the MainPage constructor to set the **ItemsSource** property:

```
public MainPage()
{
        InitializeComponent();
            c1datagrid.ItemsSource = Data.GetSubCategories(null).Take(10);
     }
```

Add code for the c1datagrid_ LoadedRowDetailsPresenter event to the **MainPage** class:

```
private void c1datagrid_ LoadedRowDetailsPresenter(object sender, C1.Silverlight.DataGrid.DataGridR
{
    if (e.Row.DetailsVisibility == Visibility.Visible)
    {
        C1.Silverlight.DataGrid.C1DataGrid detailGrid = e.DetailsElement as C1.Silverlight.DataGrid
        if (detailGrid.ItemsSource == null)
        {
            int subcategory = (e.Row.DataItem as Subcategory).ProductSubcategoryID;
            detailGrid.ItemsSource = Data.GetProducts((product) => product.Element("ProductSubcateg
        }
    }
}
```

Add code for the c1datagrid_LoadingRow event to the **MainPage** class to set the row details visibility for the first row:

```
private void c1datagrid_LoadingRow(object sender, DataGridRowEventArgs e)
{
    if (e.Row.Index == 0)
    {
        e.Row.DetailsVisibility = Visibility.Visible;
    }
}
```

Now you may run it so you may see what you've accomplished.

Summary

ComponentOne DataGrid for Silverlight makes creating Master-Detail views a breeze. This tutorial demonstrated how easily you can create a grid in your Silverlight application that allows end users to easily and effectively navigate and find the data they are looking for. Use ComponentOne DataGrid for Silverlight to take your Silverlight applications to the next level.

Download ComponentOne Studio for Silverlight