

---

ComponentOne

# DockControl for Silverlight

Copyright © 2012 ComponentOne LLC. All rights reserved.

*Corporate Headquarters*

**ComponentOne LLC**

201 South Highland Avenue  
3<sup>rd</sup> Floor  
Pittsburgh, PA 15206 • USA

**Internet:** [info@ComponentOne.com](mailto:info@ComponentOne.com)

**Web site:** <http://www.componentone.com>

**Sales**

E-mail: [sales@componentone.com](mailto:sales@componentone.com)

Telephone: 1.800.858.2739 or 1.412.681.4343 (Pittsburgh, PA USA Office)

**Trademarks**

The ComponentOne product name is a trademark and ComponentOne is a registered trademark of ComponentOne LLC. All other trademarks used herein are the properties of their respective owners.

**Warranty**

ComponentOne warrants that the original CD (or diskettes) are free from defects in material and workmanship, assuming normal use, for a period of 90 days from the date of purchase. If a defect occurs during this time, you may return the defective CD (or disk) to ComponentOne, along with a dated proof of purchase, and ComponentOne will replace it at no charge. After 90 days, you can obtain a replacement for a defective CD (or disk) by sending it and a check for \$25 (to cover postage and handling) to ComponentOne.

Except for the express warranty of the original CD (or disks) set forth here, ComponentOne makes no other warranties, express or implied. Every attempt has been made to ensure that the information contained in this manual is correct as of the time it was written. We are not responsible for any errors or omissions. ComponentOne's liability is limited to the amount you paid for the product. ComponentOne is not liable for any special, consequential, or other damages for any reason.

**Copying and Distribution**

While you are welcome to make backup copies of the software for your own use and protection, you are not permitted to make copies for the use of anyone else. We put a lot of time and effort into creating this product, and we appreciate your support in seeing that it is used by licensed users only.

This manual was produced using [ComponentOne Doc-To-Help™](#).

# Table of Contents

|  |    |
|--|----|
| ComponentOne DockControl for Silverlight .....                     | 1  |
| Installing Studio for Silverlight .....                            | 1  |
| Studio for Silverlight Setup Files .....                           | 1  |
| Using Maps Powered by Esri .....                                   | 2  |
| System Requirements .....  | 3  |
| Installing Demonstration Versions .....                            | 3  |
| Uninstalling Studio for Silverlight .....                          | 3  |
| End-User License Agreement .....                                   | 3  |
| Licensing FAQs .....   | 3  |
| What is Licensing? .....   | 3  |
| Studio for Silverlight Licensing .....                             | 4  |
| Technical Support .....  | 6  |
| Redistributable Files .....  | 6  |
| About This Documentation .....                                     | 7  |
| Silverlight Automation Overview .....                              | 7  |
| Automated UI Testing using the C1.Silverlight.Automation.dll ..... | 8  |
| The C1.Silverlight.Docking.dll Assembly .....                      | 14 |
| Studio for Silverlight Samples .....                               | 14 |
| C1.Silverlight.Docking Samples .....                               | 15 |
| ControlExplorer Sample .....                                       | 15 |
| Factories Sample .....   | 15 |
| Olympics Sample .....  | 15 |
| QuickStart Sample .....  | 16 |
| SamplesCommon Sample .....   | 16 |
| SilverTunes Sample .....   | 16 |
| StockPortfolio Sample .....  | 16 |
| Templates Sample .....   | 16 |
| Introduction to Silverlight .....                                  | 17 |
| Silverlight Resources .....  | 17 |
| Creating a New Silverlight Project .....                           | 18 |

|  |    |
|--|----|
| Theming .....                            | 21 |
| Available Themes .....                   | 21 |
| BureauBlack .....                        | 21 |
| Cosmopolitan .....                       | 22 |
| ExpressionDark .....                     | 23 |
| ExpressionLight .....                    | 23 |
| RainierOrange .....                      | 24 |
| ShinyBlue .....                          | 25 |
| WhistlerBlue .....                       | 25 |
| Custom Themes .....                      | 26 |
| Included XAML Files .....                | 26 |
| C1.Silverlight .....                     | 27 |
| C1.Silverlight.Chart .....               | 27 |
| C1.Silverlight.Chart.Editor .....        | 28 |
| C1.Silverlight.Chart3D .....             | 28 |
| C1.Silverlight.DataGrid .....            | 29 |
| C1.Silverlight.DataGrid.Filters .....    | 29 |
| C1.Silverlight.DataGrid.Ria .....        | 29 |
| C1.Silverlight.DataGrid.Summaries .....  | 29 |
| C1.Silverlight.DateTimeEditors .....     | 30 |
| C1.Silverlight.Docking .....             | 30 |
| C1.Silverlight.Extended .....            | 30 |
| C1.Silverlight.FlexGrid .....            | 31 |
| C1.Silverlight.FlexGrid.Filter .....     | 31 |
| C1.Silverlight.Gauge .....               | 31 |
| C1.Silverlight.Imaging .....             | 31 |
| C1.Silverlight.Legacy .....              | 31 |
| C1.Silverlight.Maps .....                | 32 |
| C1.Silverlight.MediaPlayer .....         | 32 |
| C1.Silverlight.OrgChart .....            | 32 |
| C1.Silverlight.OutlookBar .....          | 32 |
| C1.Silverlight.PdfViewer .....           | 32 |
| C1.Silverlight.ReportViewer .....        | 33 |
| C1.Silverlight.RichTextBox .....         | 33 |
| C1.Silverlight.RichTextBox.Toolbar ..... | 33 |
| C1.Silverlight.Schedule .....            | 33 |
| C1.Silverlight.SpellChecker .....        | 34 |

|  |    |
|--|----|
| C1.Silverlight.Theming.BureauBlack .....                         | 34 |
| C1.Silverlight.Theming.Cosmopolitan .....                        | 34 |
| C1.Silverlight.Theming.ExpressionDark.....                       | 35 |
| C1.Silverlight.Theming.ExpressionLight .....                     | 35 |
| C1.Silverlight.Theming.Office2007.....                           | 35 |
| C1.Silverlight.Theming.Office2010.....                           | 35 |
| C1.Silverlight.Theming.RainierOrange .....                       | 36 |
| C1.Silverlight.Theming.ShinyBlue .....                           | 36 |
| C1.Silverlight.Theming.WhistlerBlue.....                         | 36 |
| C1.Silverlight.TileView .....                                    | 36 |
| C1.Silverlight.Toolbar.....                                      | 36 |
| Implicit and Explicit Styles .....                               | 37 |
| Implicit Styles .....  | 37 |
| WPF and Silverlight Styling .....                                | 37 |
| Using the ImplicitStyleManager.....                              | 38 |
| Applying Themes to Controls.....                                 | 38 |
| Applying Themes to an Application .....                          | 40 |
| ComponentOne ClearStyle Technology .....                         | 42 |
| How ClearStyle Works .....                                       | 43 |
| ClearStyle Properties.....                                       | 43 |
| DockControl for Silverlight .....                                | 45 |
| DockControl for Silverlight Features .....                       | 45 |
| DockControl for Silverlight Quick Start .....                    | 45 |
| Step 1 of 3: Creating a Silverlight Application .....            | 46 |
| Step 2 of 3: Adding a C1DockTabControl with C1DockTabItems ..... | 46 |
| Step 3 of 3: Running the Application.....                        | 47 |
| XAML Quick Reference .....                                       | 49 |
| Working with DockControl for Silverlight .....                   | 50 |
| C1DockControl Elements.....                                      | 50 |
| Docking Options.....   | 51 |
| Docking Diamond and Zones .....                                  | 52 |
| DockControl for Silverlight Layout and Appearance .....          | 54 |
| Templates .....  | 54 |
| DockControl Theming.....   | 54 |
| DockControl ClearStyle Properties.....                           | 56 |
| DockControl for Silverlight Task-Based Help .....                | 57 |
| Setting the Dock Mode .....                                      | 58 |

Using Silverlight Themes ..... 59

# ComponentOne DockControl for Silverlight

Handle multiple windows in your Silverlight application with **ComponentOne DockControl™ for Silverlight**. Similar to the docking system in Microsoft Visual Studio 2008, DockControl delivers dockable, floating, and tabbed windows. You can also auto-hide sections and easily style the **DockControl**.

Controls included with **DockControl for Silverlight**:

- **C1DockControl**
- **C1DockTabControl**
- **C1DockTabItem**
- **C1DockGroup**

For a list of the latest features added to **ComponentOne Studio for Silverlight**, visit [What's New in Studio for Silverlight](#).

## Installing Studio for Silverlight

The following sections provide helpful information on installing **ComponentOne Studio for Silverlight**.

### Studio for Silverlight Setup Files

The **ComponentOne Studio for Silverlight** installation program will create the following directory: **C:\Program Files\ComponentOne\Studio for Silverlight 4.0**. This directory contains the following subdirectories:

- |             |   |
|-------------|---|
| <b>Bin</b>  | Contains copies of ComponentOne binaries (DLLs, EXEs, design-time assemblies).                    |
| <b>Help</b> | Contains documentation for all Studio components and other useful resources including XAML files. |

### Samples

Samples for the product are installed in the **ComponentOne Samples** folder by default. The path of the ComponentOne Samples directory is slightly different on Windows XP and Windows Vista/Windows 7 machines:

**Windows XP path:** C:\Documents and Settings\\My Documents\ComponentOne Samples\Studio for Silverlight 4.0

**Windows Vista and Windows 7 path:** C:\Users\\Documents\ComponentOne Samples\Studio for Silverlight 4.0

See the [Studio for Silverlight Samples](#) (page 14) topic for more information about each sample.

### Esri Maps

Esri® files are installed with **ComponentOne Studio for Silverlight**, **ComponentOne Studio for WPF**, and **ComponentOne Studio for Windows Phone** by default to the following folders:

32-bit machine : C:\Program Files\ESRI SDKs\\<version number>

64-bit machine: C:\Program Files (x86)\ESRI SDKs\<Platform>\<version number>

Files are provided for multiple languages, including: English, German (de), Spanish (es), French (fr), Italian (it), Japanese (ja), Portuguese (pt-BR), Russian (ru) and Chinese (zh-CN).

See [Using Maps Powered by Esri](#) (page 2) or visit the Esri website at <http://www.esri.com> for additional information.

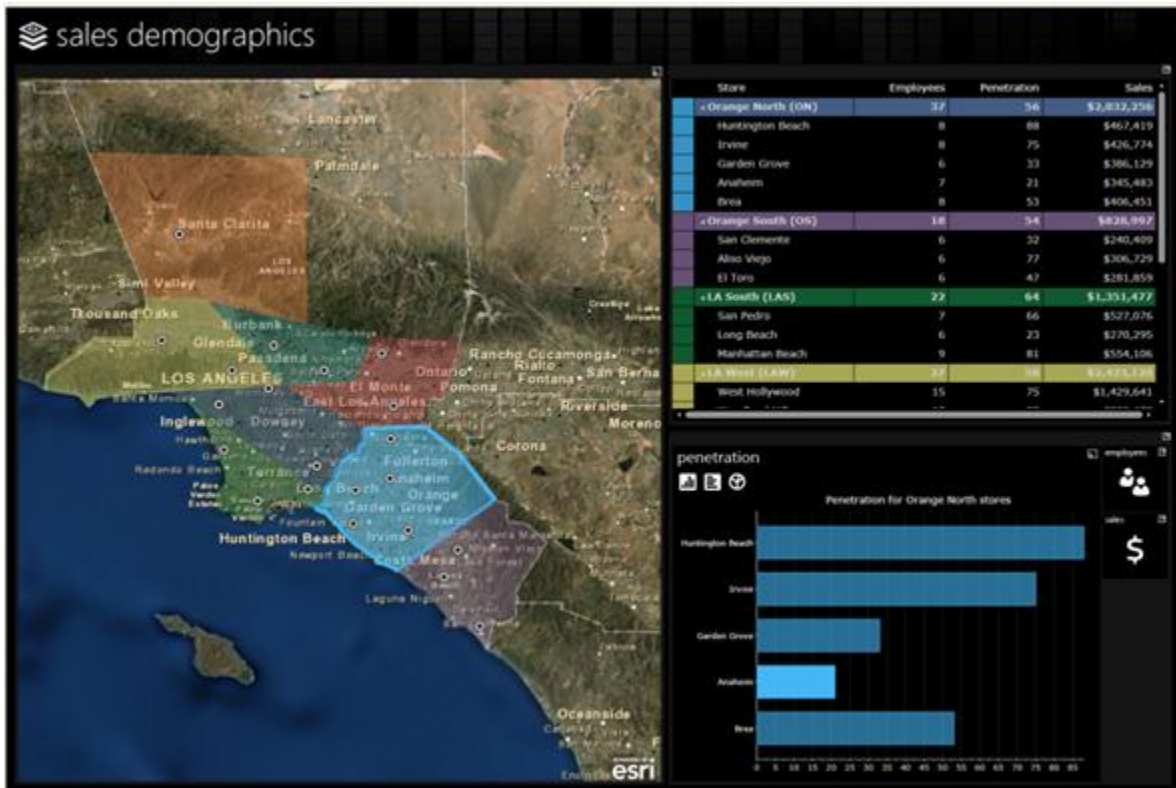
## Using Maps Powered by Esri

Easily transform GIS data into business intelligence with controls for Silverlight, WPF, and Windows Phone powered by Esri® software.

By using the ComponentOne award-winning UI controls, you'll have the tools you need to seamlessly create rich, map-enabled user interfaces.

Benefits of Maps powered by Esri:

- Esri knows maps: Esri is the leading online map and GIS provider.
- Maps are technical: Using maps within your application is a very technical thing, so you don't want to take your chance using anyone but the best.
- Company of choice: Esri is the company of choice of many top companies and government agencies.
- Fulfill any developers' mapping needs: Esri mapping tools are flexible and will fill the needs of any mapping solution.



Esri Map Example

There are no additional charges for using the Esri maps included with ComponentOne products. Simply create a free online account at <http://www.arcgisonline.com> to start taking advantage of the Esri map controls. Esri

licensing terms can be found in our Licensing Information and End User Licensing Agreement at <http://www.componentone.com/SuperPages/Licensing/>.

To learn more about Esri and Esri maps, please visit Esri at <http://www.esri.com>. There you will find detailed support, including [documentation](#), [forums](#), [samples](#), and much more.

See the [Studio for Silverlight Setup Files](#) (page 1) topic for more information on the Esri files installed with this product.

## System Requirements

System requirements for **ComponentOne Studio for Silverlight** include the following:

- Microsoft Silverlight 4.0 or later
- Microsoft Visual Studio 2008 or later

## Installing Demonstration Versions

If you wish to try **ComponentOne Studio for Silverlight** and do not have a serial number, follow the steps through the installation wizard and use the default serial number.

The only difference between unregistered (demonstration) and registered (purchased) versions of our products is that the registered version will stamp every application you compile so a ComponentOne banner will not appear when your users run the applications.

## Uninstalling Studio for Silverlight

To uninstall **ComponentOne Studio for Silverlight**:

1. Open the **Control Panel** and select **Add or Remove Programs** (XP) or **Programs and Features** (Windows 7/Vista).
2. Select **ComponentOne Studio for Silverlight 4.0** and click the **Remove** button.
3. Click **Yes** to remove the program.

## End-User License Agreement

All of the ComponentOne licensing information, including the ComponentOne end-user license agreements, frequently asked licensing questions, and the ComponentOne licensing model, is available online at <http://www.componentone.com/SuperPages/Licensing/>.

## Licensing FAQs

The **ComponentOne Studio for Silverlight** product is a commercial product. It is not shareware, freeware, or open source. If you use it in production applications, please purchase a copy from our Web site or from the software reseller of your choice.

This section describes the main technical aspects of licensing. It may help the user to understand and resolve licensing problems he may experience when using ComponentOne products.

### What is Licensing?

Licensing is a mechanism used to protect intellectual property by ensuring that users are authorized to use software products.

Licensing is not only used to prevent illegal distribution of software products. Many software vendors, including ComponentOne, use licensing to allow potential users to test products before they decide to purchase them.

Without licensing, this type of distribution would not be practical for the vendor or convenient for the user. Vendors would either have to distribute evaluation software with limited functionality, or shift the burden of managing software licenses to customers, who could easily forget that the software being used is an evaluation version and has not been purchased.

## Studio for Silverlight Licensing

Licensing for **ComponentOne Studio for Silverlight** is similar to licensing in other ComponentOne products but there are a few differences to note.

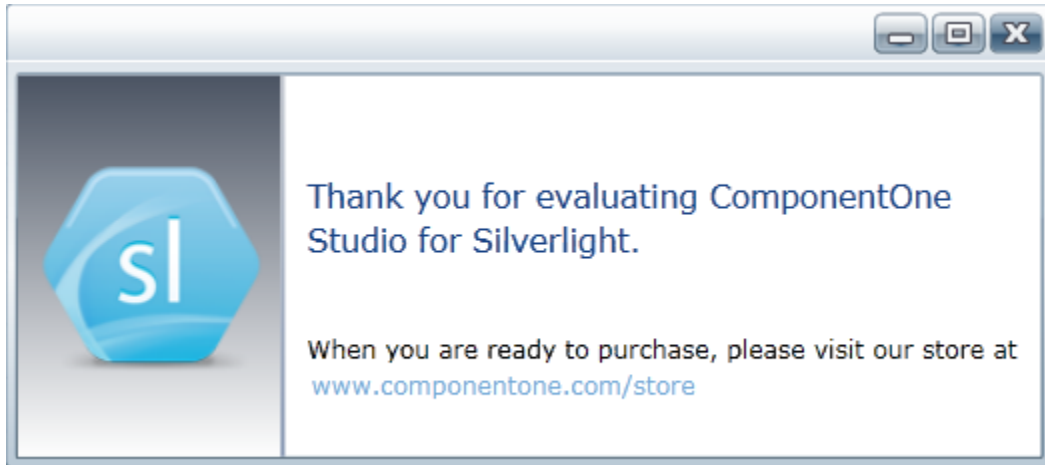
Initially licensing is handled similarly to other ComponentOne products. When a user decides to purchase a product, he receives an installation program and a Serial Number. During the installation process, the user is prompted for the serial number that is saved on the system.

In **ComponentOne Studio for Silverlight**, when a control is dropped on a form, a license nag dialog box appears one time. The nag screen appears similar to the following image:

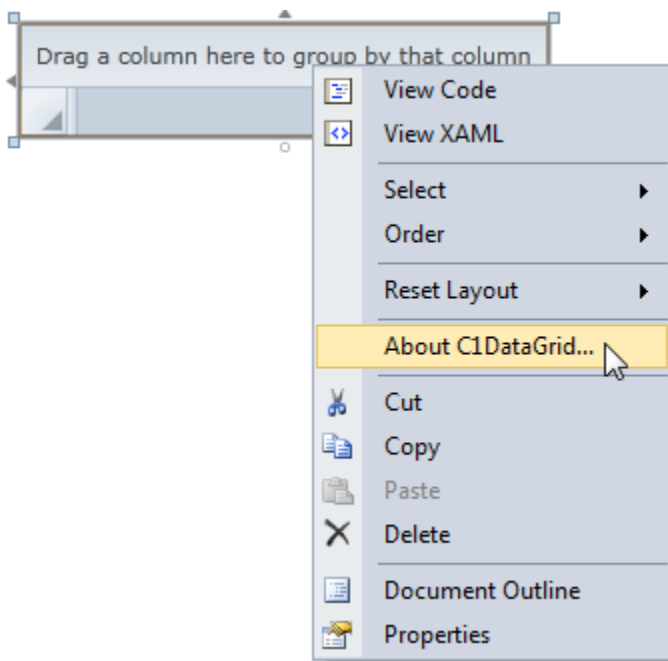


The **About** dialog box displays version information, online resources, and (if the control is unlicensed) buttons to purchase, activate, and register the product.

All ComponentOne products are designed to display licensing information at run time if the product is not licensed. None will throw licensing exceptions and prevent applications from running. Each time an unlicensed Silverlight application is run; end-users will see the following pop-up dialog box:



To stop this message from appearing, enter the product's serial number by clicking the **Activate** button on the **About** dialog box of any ComponentOne product, if available, or by rerunning the installation and entering the serial number in the licensing dialog box. To open the **About** dialog box, right-click the control and select the **About** option:



Note that when the user modifies any property of a ComponentOne Silverlight control in Visual Studio or Blend, the product will check if a valid license is present. If the product is not currently licensed, an attached property will be added to the control (the **C1NagScreen.Nag** property). Then, when the application executed, the product will check if that property is set, and show a nag screen if the **C1NagScreen.Nag** property is set to **True**. If the user has a valid license the property is not added or is just removed.

One important aspect of this of this process is that the user should manually remove all instances of **c1:C1NagScreen.Nag="true"** in the XAML markup in all files after registering the license (or re-open all the files

that include ComponentOne controls in any of the editors). This will ensure that the nag screen does not appear when the application is run.

## Technical Support

ComponentOne offers various support options. For a complete list and a description of each, visit the ComponentOne Web site at <http://www.componentone.com/SuperProducts/SupportServices/>.

Some methods for obtaining technical support include:

- **Online Resources**  
ComponentOne provides customers with a comprehensive set of technical resources in the form of FAQs, [samples and videos](#), Version Release History, searchable Knowledge base, searchable Online Help and more. We recommend this as the first place to look for answers to your technical questions.
- **Online Support via our Incident Submission Form**  
This online support service provides you with direct access to our Technical Support staff via an [online incident submission form](#). When you submit an incident, you'll immediately receive a response via e-mail confirming that you've successfully created an incident. This email will provide you with an Issue Reference ID and will provide you with a set of possible answers to your question from our Knowledgebase. You will receive a response from one of the ComponentOne staff members via e-mail in 2 business days or less.
- **Product Forums**  
ComponentOne's [product forums](#) are available for users to share information, tips, and techniques regarding ComponentOne products. ComponentOne developers will be available on the forums to share insider tips and technique and answer users' questions. Please note that a ComponentOne User Account is required to participate in the ComponentOne Product Forums.
- **Installation Issues**  
Registered users can obtain help with problems installing ComponentOne products. Contact technical support by using the [online incident submission form](#) or by phone (412.681.4738). Please note that this does not include issues related to distributing a product to end-users in an application.
- **Documentation**  
Microsoft integrated ComponentOne documentation can be installed with each of our products, and documentation is also available online. If you have suggestions on how we can improve our documentation, please email the [Documentation team](#). Please note that e-mail sent to the [Documentation team](#) is for documentation feedback only. [Technical Support](#) and [Sales](#) issues should be sent directly to their respective departments.

**Note:** You must create a ComponentOne Account and register your product with a valid serial number to obtain support using some of the above methods.

## Redistributable Files

**ComponentOne Studio for Silverlight** is developed and published by ComponentOne LLC. You may use it to develop applications in conjunction with Microsoft Visual Studio or any other programming environment that enables the user to use and integrate the control(s). You may also distribute, free of royalties, the following Redistributable Files with any such application you develop to the extent that they are used separately on a single CPU on the client/workstation side of the network:

- C1.Silverlight.dll
- C1.Silverlight.Docking.dll
- C1.Silverlight.Docking.5.dll
- C1.Silverlight.Automation.dll

- C1.Silverlight.Automation.5.dll

Site licenses are available for groups of multiple developers. Please contact [Sales@ComponentOne.com](mailto:Sales@ComponentOne.com) for details.

## About This Documentation

### Acknowledgements

Microsoft, Windows, Windows Vista, and Visual Studio, and Silverlight, are either registered trademarks or trademarks of Microsoft Corporation in the United States and/or other countries. Firefox is a registered trademark of the Mozilla Foundation. Safari is a trademark of Apple Inc., registered in the U.S. and other countries.

Esri is a registered trademark of Environmental Systems Research Institute, Inc. (Esri) in the United States, the European Community, or certain other jurisdictions.

### ComponentOne

If you have any suggestions or ideas for new features or controls, please call us or write:

*Corporate Headquarters*

#### ComponentOne LLC

201 South Highland Avenue  
3rd Floor  
Pittsburgh, PA 15206 • USA  
412.681.4343  
412.681.4384 (Fax)

<http://www.componentone.com>

### ComponentOne Doc-To-Help

This documentation was produced using [ComponentOne Doc-To-Help® Enterprise](#).

## Silverlight Automation Overview

The **C1.Silverlight.Automation.dll** is a set of Automation Peer classes for the ComponentOne Silverlight controls.

Each AutomationPeer exposes a matching control class to the Microsoft automation framework so the control can be used in automated UI tests.

For more information about the Microsoft Automation framework, see <http://msdn.microsoft.com/en-us/library/ms747327.aspx>

Below is the list of supported automation patterns for the 2012 v1 release.

| ComponentOne Silverlight Automation Peer | Supported Automation Patterns              |
|--|--|
| C1TreeView                               | Selection                                  |
| C1TreeViewItem                           | ExpandCollapse, SelectionItem, ScrollItem  |
| C1TabControl                             | Selection                                  |
| C1TabItem                                | SelectionItem, ScrollItem                  |
| C1DockTabControl                         | <b>*partially*</b> Window, Transform, Dock |
| C1Book                                   |  |
| C1BookZone                               | Invoke                                     |
| C1FlexComboBox                           | ExpandCollapse, Selection                  |

|                 |   |
|-----------------|---|
| C1FlexGridCell  | Invoke, Value, SelectionItem, ScrollItem, TableItem, GridItem |
| C1FlexGridPanel |   |
| C1FlexGridRow   | ExpandCollapse, ScrollItem, SelectionItem                     |
| C1Gauge         | Value, RangeValue   |
| C1LinearGauge   | Value, RangeValue   |
| C1Knob          | Value, RangeValue   |
| C1OutlookBar    | ExpandCollapse, Selection                                     |
| C1TileView      | Selection   |
| C1TileViewItem  | SelectionItem, ExpandCollapse, ScrollItem                     |

For more info about automation patterns, see <http://msdn.microsoft.com/en-us/library/ms752362.aspx>.

#### Using the C1.Silverlight.Automation.dll

To start using the **C1.Silverlight.Automation.dll** in a Silverlight application, simply add a reference to it in the project.

### Automated UI Testing using the C1.Silverlight.Automation.dll

Let's add unit testing for the **OutlookBarSamples** from **ComponentOne Studio for Silverlight** samples as an example.

---

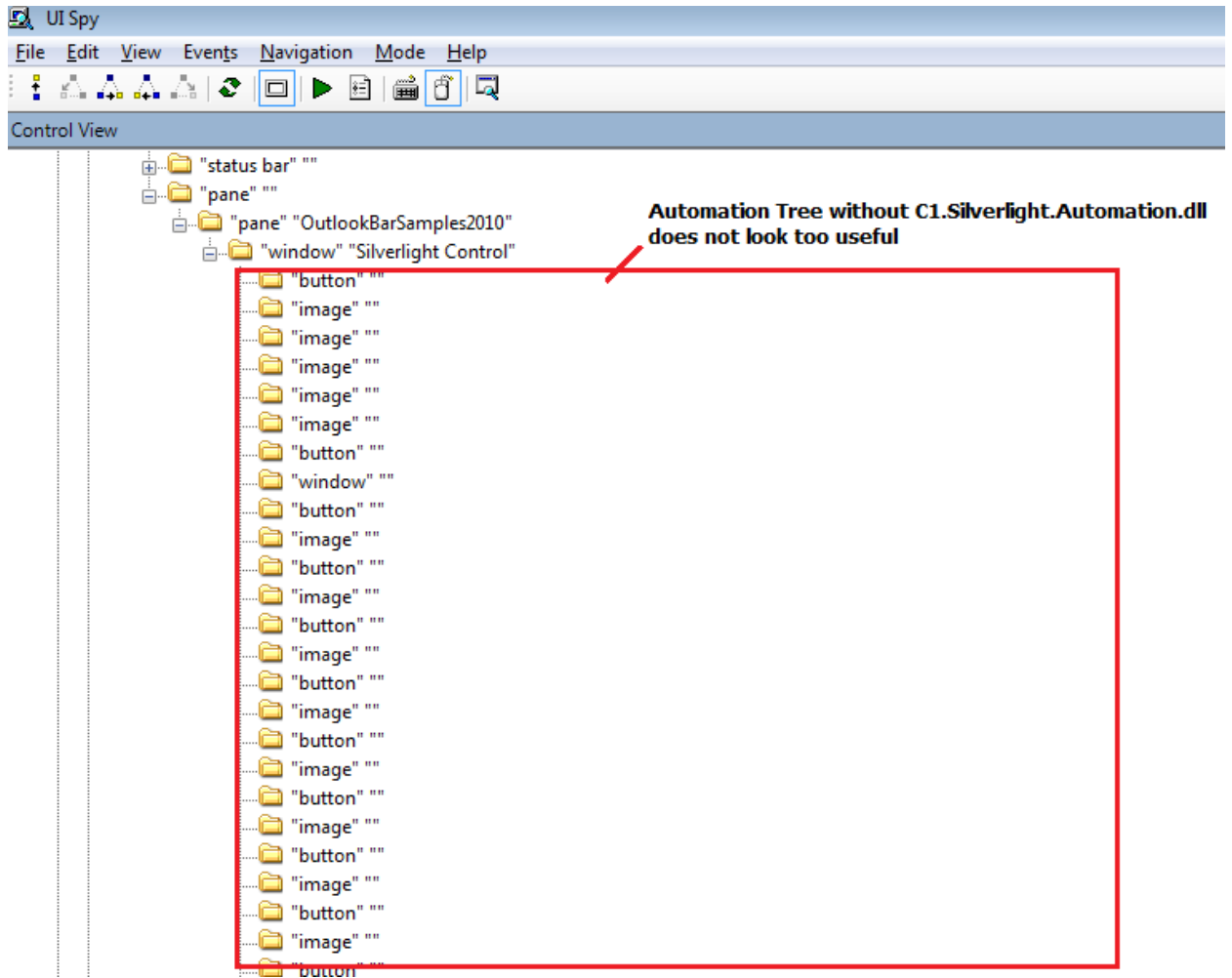
**NOTE:** ComponentOne samples are installed, by default, in the **ComponentOne Samples** folder in C:\Documents and Settings\\My Documents\ComponentOne Samples\Studio for Silverlight (**Windows XP**) or C:\Users\\Documents\ComponentOne Samples\Studio for Silverlight (**Windows Vista and Windows 7**).

---

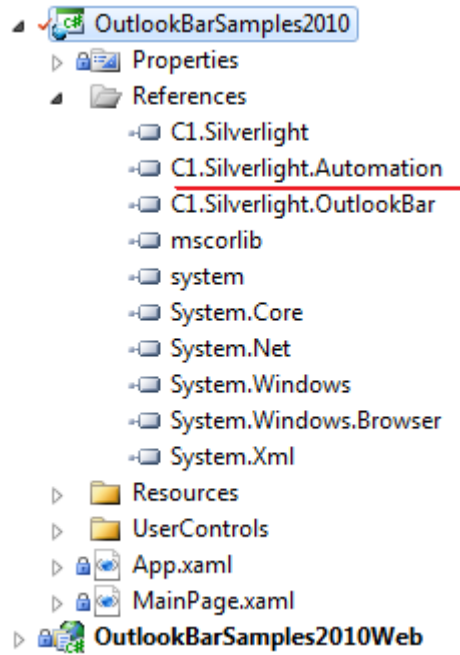
This first step shows what we see without using the C1.Silverlight.Automation.dll.

1. Start the **OutlookBarSamples** without any modifications and explore the automation tree of the application using **UI Spy** or a similar tool.

The tree looks like an unorganized list of images and buttons. It does not seem to be useful for writing tests.

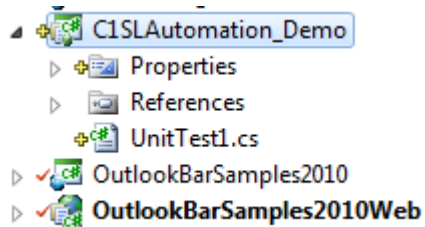


2. Add a reference to the C1.Silverlight.Automation.dll to your project.



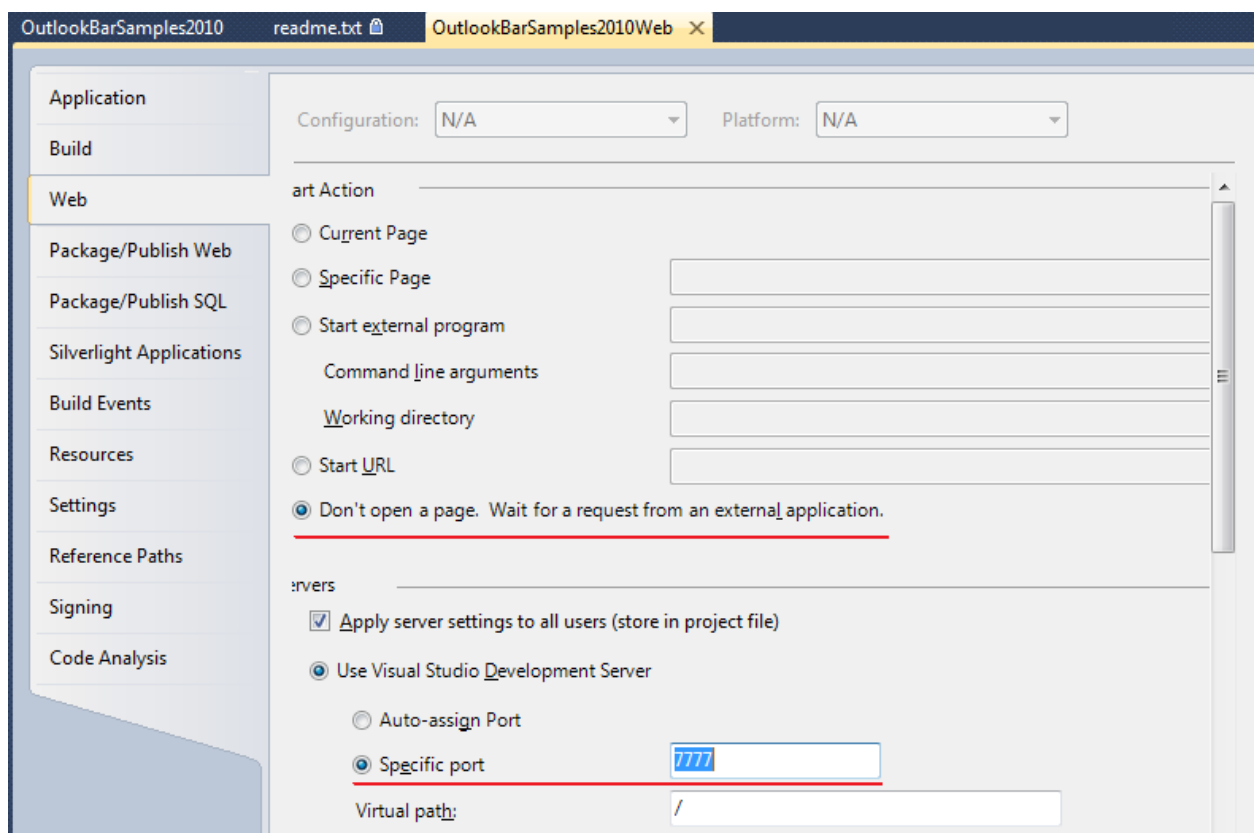
3. Start the application and explore the automation tree:





5. Set up the solution:

- Set **OutlookBarSamples2010Web** as the startup project.
- Specify the project properties as in the following example:



6. Add a CodedUITest to the test project.

7. Write some unit tests. Find automation elements and use their patterns.

```
[TestMethod]
public void OutlookBarCollapseExpand()
{
    BrowserWindow.CurrentBrowser = "IE";
    // Run Browser
    BrowserWindow win = BrowserWindow.Launch("about:blank");
}
```

```

        System.Diagnostics.Process process = SearchIEProcess("Blank
Page");
        try
        {

            win.NavigateToUrl(new Uri("http://localhost:7777"));
            // Wait some time to complete
            System.Threading.Thread.Sleep(5000);
            AutomationElement mainElement =
System.Windows.Automation.AutomationElement.FromHandle(process.MainWindowHa
ndle);

            AutomationElement outlookBar =
mainElement.FindFirst(TreeScope.Descendants, new
PropertyCondition(AutomationElement.ClassNameProperty, "ClOutlookBar"));
            Assert.IsNotNull(outlookBar, "Cannot find control.");

            object pattern = null;

            outlookBar.TryGetCurrentPattern(ExpandCollapsePattern.Pattern, out
pattern);

            Assert.IsNotNull(pattern, "Cannot find
ExpandCollapsePattern.");
            ExpandCollapsePattern expandCollapsePattern =
(ExpandCollapsePattern)pattern;

            bool collapsed =
expandCollapsePattern.Current.ExpandCollapseState ==
ExpandCollapseState.Collapsed;
            Assert.IsFalse(collapsed, "OutlookBar should be expanded on
start");

            expandCollapsePattern.Collapse();
            collapsed =
expandCollapsePattern.Current.ExpandCollapseState ==
ExpandCollapseState.Collapsed;
            Assert.IsTrue(collapsed, "OutlookBar should be collapsed
now");

            var items = mainElement.FindAll(TreeScope.Descendants, new
PropertyCondition(AutomationElement.ClassNameProperty, "ClOutlookItem"));
            Assert.AreEqual(5, items.Count, "5 OutlookItems expected");

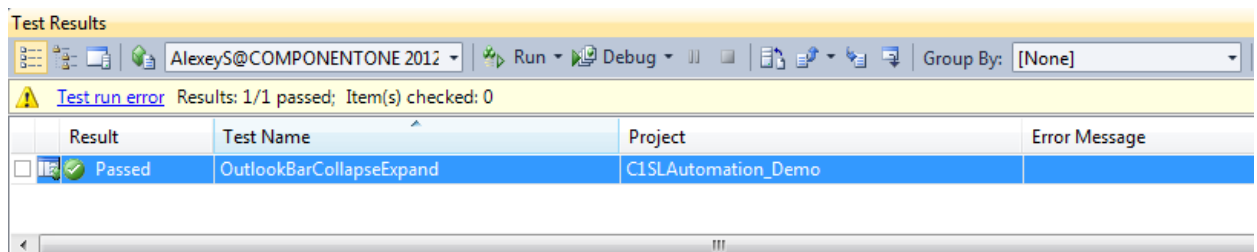
```

```

        System.Threading.Thread.Sleep(2000);
    }
    finally
    {
        // Close browser
        process.CloseMainWindow();
    }
}

```

8. Run the tests.



## The C1.Silverlight.Docking.dll Assembly

The **C1.Silverlight.Docking** assembly contains basic docking panels and controls.

### Main Classes

The following main classes are included in the **C1.Silverlight.Docking.dll** assembly:

- **C1DockControl**: Similar to the docking system in Microsoft Visual Studio 2008, DockControl delivers dockable, floating, and tabbed windows. You can also auto-hide sections and easily style the DockControl.

## Studio for Silverlight Samples

If you just installed **ComponentOne Studio for Silverlight**, open Visual Studio 2008 and load the **Samples.sln** solution located in the **C:\Documents and Settings\<username>\My Documents\ComponentOne Samples\Studio for Silverlight** or **C:\Users\<username>\Documents\ComponentOne Samples\Studio for Silverlight** folder. This solution contains all the samples that ship with this release. Each sample has a readme.txt file that describes it and two projects named as follows:

|                              |  |
|------------------------------|--|
| <b>&lt;SampleName&gt;</b>    | Silverlight project (client-side project)                                |
| <b>&lt;SampleName&gt;Web</b> | ASP.NET project that hosts the Silverlight project (server-side project) |

To run a sample, right-click the **<SampleName>Web** project in the Solution Explorer, select **Set as Startup Project**, and press F5.

The following topics, organized by folder, describe each of the included samples.

## **C1.Silverlight.Docking Samples**

The following samples are installed in the **C1.Silverlight.Docking** folder in the samples directory by default.

### ***Blend Sample***

The **Blend** sample is installed in the **C1.Silverlight.Docking\Blend** folder in the samples directory by default.

This sample demonstrates the controls belonging to the C1.Silverlight.Docking.dll assembly and uses the controls, including docking and tab controls, to create a Microsoft Expression Blend-like environment.

### ***C1Docking\_Demo Sample***

The **C1Docking\_Demo** sample is installed in the **C1.Silverlight.Docking\C1Docking\_Demo** folder in the samples directory by default.

This sample demonstrates the controls belonging to the C1.Silverlight.Docking.dll assembly and uses the controls, including docking and tab controls, to demonstrate docking windows that fill the application window, that slide out to show content, and that display content in a floating window.

### ***VSDev9 Sample***

The **VSDev9** sample is installed in the **C1.Silverlight.Docking\VSDev9** folder in the samples directory by default.

This sample demonstrates the controls belonging to the C1.Silverlight.Docking.dll assembly and uses the controls, including docking and tab controls, to create a Visual Studio-like environment.

### **ControlExplorer Sample**

The **ControlExplorer** sample is installed in the **ControlExplorer** folder in the samples directory.

This sample displays a tree with all C1.Silverlight controls and demonstrates them in action. The application has two panes. The left pane contains a tree that lists all controls in the C1.Silverlight library. Select a control on the tree and an instance appears on the right pane, along with a few properties that you can modify to see how they affect the control.

The projects show all controls in the C1.Silverlight library, and shows how to use reflection to create controls and interact with them dynamically. The project is driven by an XML resource file that lists the controls that should be exposed by the sample. The DataGrid sample uses a data source that is a DataSet stored as a zip-compressed embedded resource.

### **Factories Sample**

The **Factories** sample is installed in the **Factories** folder in the samples directory.

This sample demonstrates how to use **C1Maps** to show production and distribution information from a manufacturer. There are three types of location marked in the map: factories, offices and stores. Each has a small icon providing related information, click on it to view an expanded version. Stores are only visible by zooming in near an office. The user can position new locations by dragging them from the top-right toolbar to the map.

### **Olympics Sample**

The **Olympics** sample is installed in the **Olympics** folder in the samples directory.

This sample allows you to check out the performance of any country in the Olympic Games. The sample was built during the 2008 Olympic Games in Beijing, and got a lot of traffic during the games. The sample used a Web service to retrieve the number of medals won by each country in real time, and then displayed that information in three different ways: map, chart, and grid.

## QuickStart Sample

The **Factories** sample is installed in the **QuickStart** folder in the samples directory.

This sample demonstrates how to instantiate and use the several of the controls in **ComponentOne Studio for Silverlight**.

## SamplesCommon Sample

The **SamplesCommon** project is installed in the **SamplesCommon\SamplesCommon** folder in the samples directory. This sample project includes elements that are incorporated in other sample projects.

## SilverTunes Sample

The **SilverTunes** sample is installed in the **SilverTunes** folder in the samples directory.

This sample showcases a Silverlight implementation of an 'iTunes' like application. The sample loads artist, album, and song information from a database. It then shows the album covers in a carousel, with a rotating effect and reflection. The user can search by artist, album, and song. The sample includes a few MP3 files so the user can play some of the songs. The MP3 files included in the sample are provided for demonstration purposes only and are truncated to protect the copyright owners.

## StockPortfolio Sample

The **StockPortfolio** sample is installed in the **StockPortfolio** folder in the samples directory.

This sample demonstrates a stock portfolio application. This sample was created for the ReMIX 07 in Boston, to demonstrate a business application in Silverlight. When it starts, the sample loads the stock portfolio from isolated storage (or creates a new one when run for the first time). Once the portfolio is loaded, the application retrieves real-time stock quotes using a web service and shows the portfolio information in a grid. The grid shows the personal portfolio information (stocks, quantities, price paid), real time stock information (last traded price, day high and low), and some calculated data (current market value, gain/loss).

The user can add new stocks to the portfolio or remove existing ones. When the application exits, the current portfolio information is saved to isolated storage so it is available next time the application runs. The user can also see the historical data for companies in the portfolio. Double-clicking a symbol on the grid shows a composite chart with detailed information on the top and a zoom chart below. The user can slide and expand the zoom window in the bottom chart to see details on the top chart.

## Templates Sample

The **Templates** sample is installed in the **Templates** folder in the samples directory.

This sample shows how to use Data Templates to determine how data is displayed in controls. The sample creates a list of items and applies the same data template to two controls of different types (a **ListBox** and a **ComboBox**).

# Introduction to Silverlight

The following topics detail information about getting started with Silverlight, including Silverlight resources, and general information about templates and deploying Silverlight files.

## Silverlight Resources

This help file focuses on **ComponentOne Studio for Silverlight**. For general help on getting started with Silverlight, we recommend the following resources:

- <http://www.silverlight.net>  
The official Silverlight site, with many links to downloads, samples, tutorials, and more.
- <http://silverlight.net/learn/tutorials.aspx>  
Silverlight tutorials by Jesse Liberty. Topics covered include:
  - Tutorial 1: Silverlight User Interface Controls
  - Tutorial 2: Data Binding
  - Tutorial 3: Displaying SQL Database Data in a DataGrid using LINQ and WCF
  - Tutorial 4: User Controls
  - Tutorial 5: Styles, Templates and Visual State Manager
  - Tutorial 6: Expression Blend for Developers
  - Tutorial 7: DataBinding & DataTemplates Using Expression Blend
  - Tutorial 8: Multi-page Applications
  - Tutorial 9: ADO.NET DataEntities and WCF Feeding a Silverlight DataGrid
  - Tutorial 10: Hyper-Video
- <http://timheuer.com/blog/articles/getting-started-with-silverlight-development.aspx>  
Silverlight tutorials by Tim Heuer. Topics covered include:
  - Part 1: Really getting started – the tools you need and getting your first Hello World
  - Part 2: Defining UI Layout – understanding layout and using Blend to help
  - Part 3: Accessing data – how to get data from where
  - Part 4: Binding the data – once you get the data, how can you use it?
  - Part 5: Integrating additional controls – using controls that aren't a part of the core
  - Part 6: Polishing the UI with styles and templates
  - Part 7: Taking the application out-of-browser
- <http://weblogs.asp.net/scottgu/pages/silverlight-posts.aspx>  
Scott Guthrie's Silverlight Tips, Tricks, Tutorials and Links Page. A useful resource, this page links to several tutorials and samples.
- <http://weblogs.asp.net/scottgu/archive/2008/02/22/first-look-at-silverlight-2.aspx>  
An excellent eight-part tutorial by Scott Guthrie, covering the following topics:

- Part 1: Creating "Hello World" with Silverlight 2 and VS 2008
- Part 2: Using Layout Management
- Part 3: Using Networking to Retrieve Data and Populate a DataGrid
- Part 4: Using Style Elements to Better Encapsulate Look and Feel
- Part 5: Using the ListBox and DataBinding to Display List Data
- Part 6: Using User Controls to Implement Master/Details Scenarios
- Part 7: Using Templates to Customize Control Look and Feel
- Part 8: Creating a Digg Desktop Version of our Application using WPF
- <http://blogs.msdn.com/corrinab/archive/2008/03/11/silverlight-2-control-skins.aspx>  
A practical discussion of skinning Silverlight controls and applications by Corrina Barber.

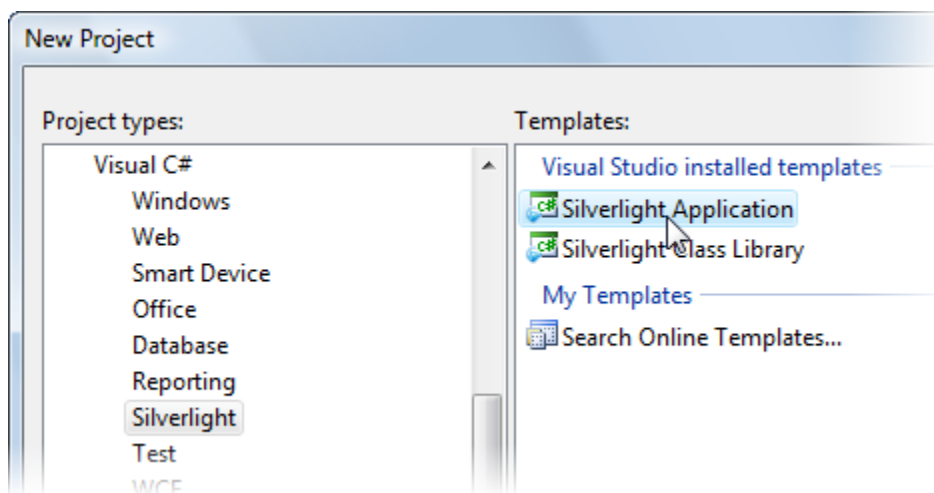
## Creating a New Silverlight Project

The following topic details how to create a new Silverlight project in Microsoft Visual Studio 2008 and in Microsoft Expression Blend 3.

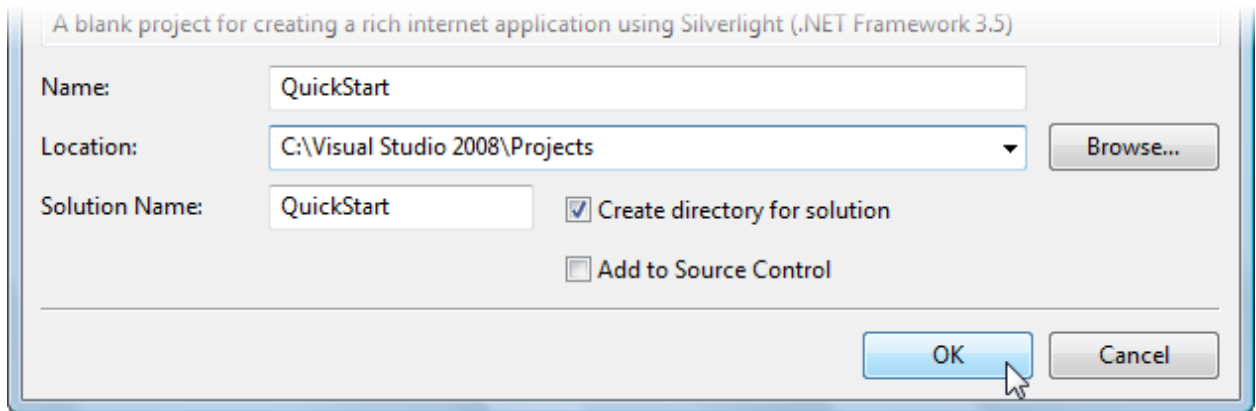
### In Visual Studio 2008

Complete the following steps to create a new Silverlight project in Microsoft Visual Studio 2008:

1. Select **File | New | Project** to open the **New Project** dialog box in Visual Studio 2008.
2. In the **Project types** pane, expand either the **Visual Basic** or **Visual C#** node and select **Silverlight**.
3. Choose **Silverlight Application** in the **Templates** pane.

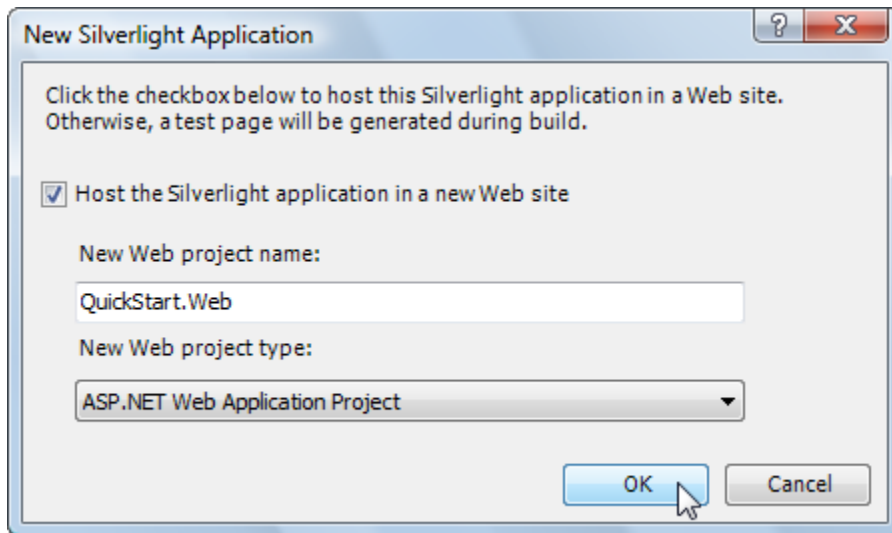


4. Name the project, specify a location for the project, and click **OK**.



Next, Visual Studio will prompt you for the type of hosting you want to use for the new project.

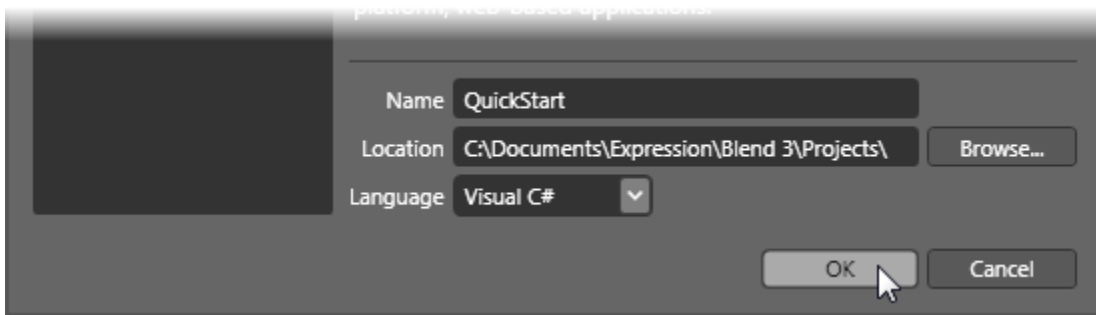
5. In the **New Silverlight Application** dialog box, select **OK** to accept the default name and options and to create the project.



#### In Expression Blend 4

Complete the following steps to create a new Silverlight project in Microsoft Expression Blend 4:

1. Select **File | New Project** to open the **New Project** dialog box in Blend 4.
2. In the **Project types** pane, click the **Silverlight** node.
3. In the right pane, choose **Silverlight Application + Website** in the **Templates** pane to create a project with an associated Web site.
4. Name the project, specify a location for the project, choose a language (**Visual C#** or **Visual Basic**), and click **OK**.



Your new project will be created.

### The Project

The solution you just created will contain two projects, **YourProject** and **YourProject.Web**:

- **YourProject**: This is the Silverlight application proper. It will produce a XAP file that gets downloaded to the client and runs inside the Silverlight plug-in.
- **YourProject.Web**: This is the host application. It runs on the server and provides support for the Silverlight application.

# Theming

One of the main advantages to using Silverlight is the ability to change the style or template of any control. Controls are "lookless" with fully customizable user interfaces and the ability to use built-in and custom themes. Themes allow you to customize the appearance of controls and take advantage of Silverlight's XAML-based styling. The following topics introduce you to styling Silverlight controls with themes.

You can customize WPF and Microsoft Silverlight controls by creating and modifying control templates and styles. This results in a unique and consistent look for your application.

Templates and styles define the pieces that make up a control and the default behavior of the control, respectively. You can create templates and styles by making copies of the original styles and templates for a control. Modifying templates and styles is an easy way to essentially make new controls in Design view of Microsoft Expression Blend, without having to use code. The following topics provide a detailed comparison of styles and templates to help you decide whether you want to modify the style or template of a control, or both. The topics also discuss the built-in themes available in **ComponentOne Studio for Silverlight**.

## Available Themes

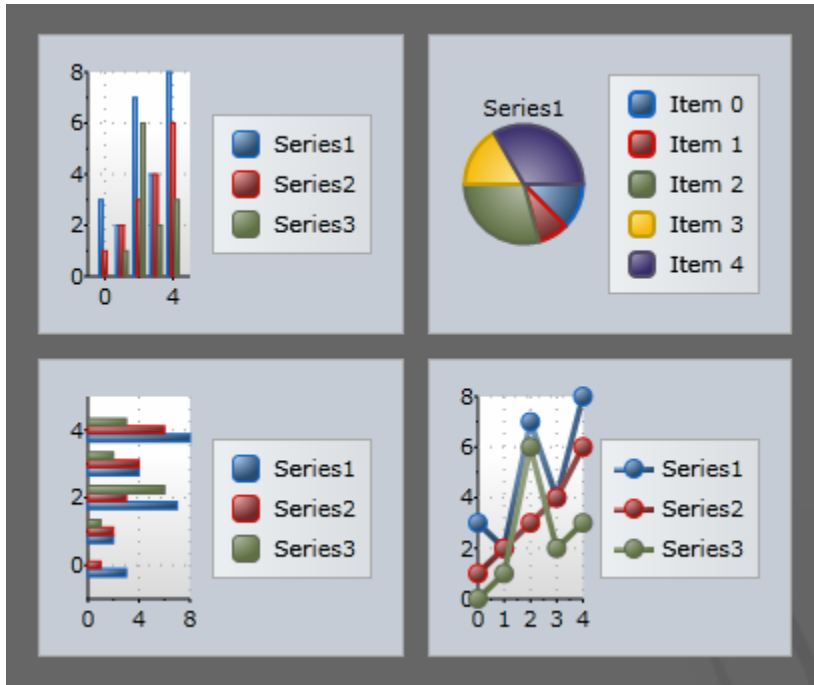
**ComponentOne Studio for Silverlight** includes several theming options, and several built-in Silverlight Toolkit themes including:

- BureauBlack
- Cosmopolitan
- ExpressionDark
- ExpressionLight
- RainierOrange
- ShinyBlue
- WhistlerBlue

Each of these themes is based on themes in the Silverlight Toolkit and installed in its own assembly in the **Studio for Silverlight** installation directory. The following topics detail each built-in theme.

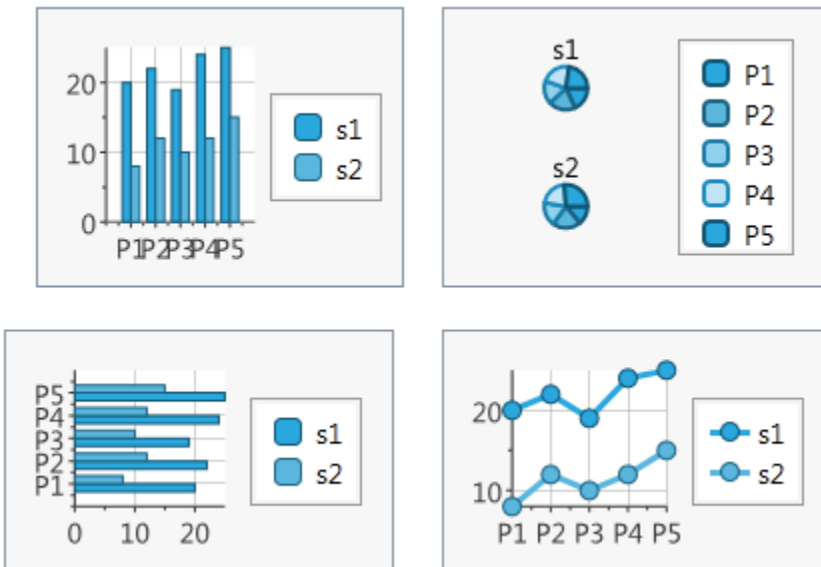
### BureauBlack

The BureauBlack theme is a dark colored theme similar to the Microsoft Bureau Black theme included in the Silverlight Toolkit. The BureauBlack theme appears similar to the following when applied to the **ComponentOne Studio for Silverlight** charting controls:



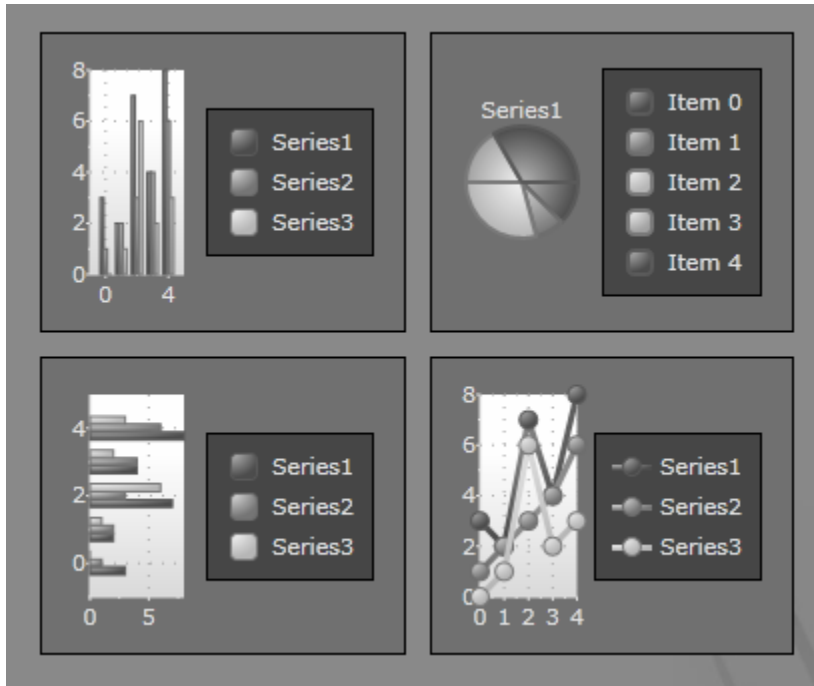
## Cosmopolitan

The Cosmopolitan theme is a modern, clean UI theme based on the Microsoft Cosmopolitan theme, which is included in the Silverlight Toolkit. For example, the theme appears similar to the following when applied to the **ComponentOne Studio for Silverlight** charting controls:



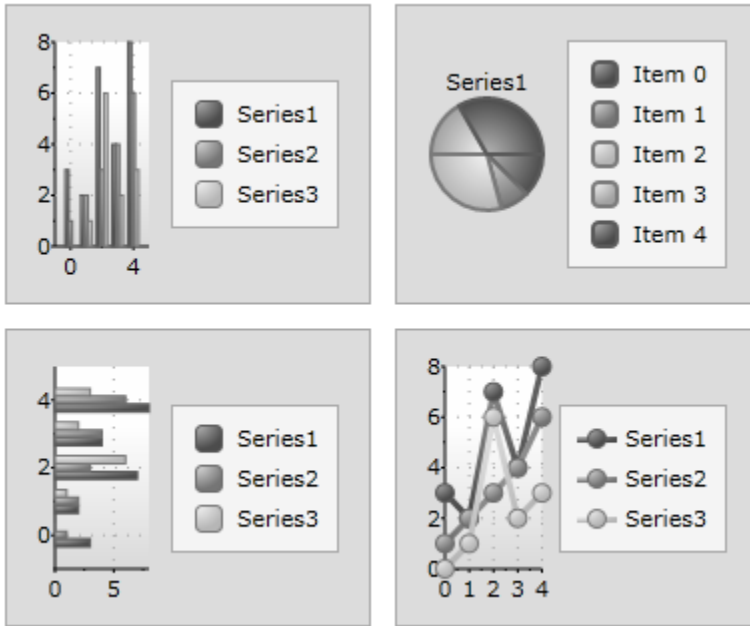
## ExpressionDark

The ExpressionDark theme is a grayscale theme based on the Microsoft Expression Dark theme, which is included in the Silverlight Toolkit. For example, the theme appears similar to the following when applied to the **ComponentOne Studio for Silverlight** charting controls:



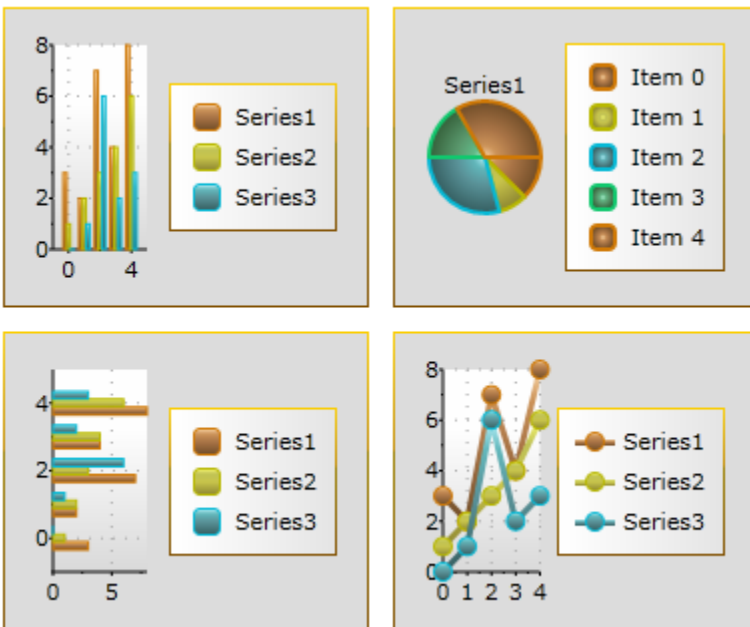
## ExpressionLight

The ExpressionLight theme is a grayscale theme based on the Microsoft Expression Light theme, which is included in the Silverlight Toolkit. For example, the theme appears similar to the following when applied to the **ComponentOne Studio for Silverlight** charting controls:



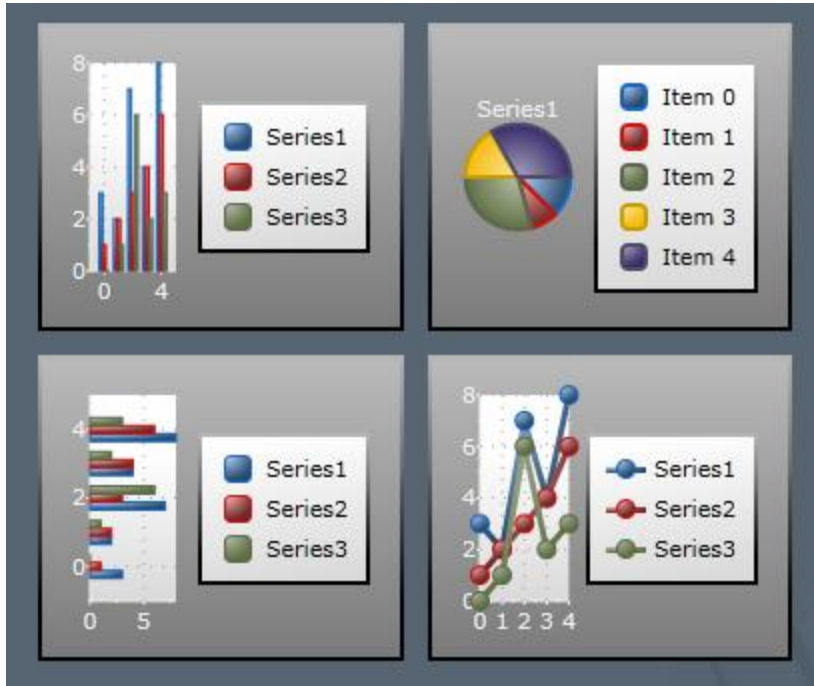
## RainierOrange

The RainierOrange theme is an orange-based theme similar to the Microsoft Rainer Orange theme, which is included in the Silverlight Toolkit. The RainierOrange theme appears similar to the following when applied to the **ComponentOne Studio for Silverlight** charting controls:



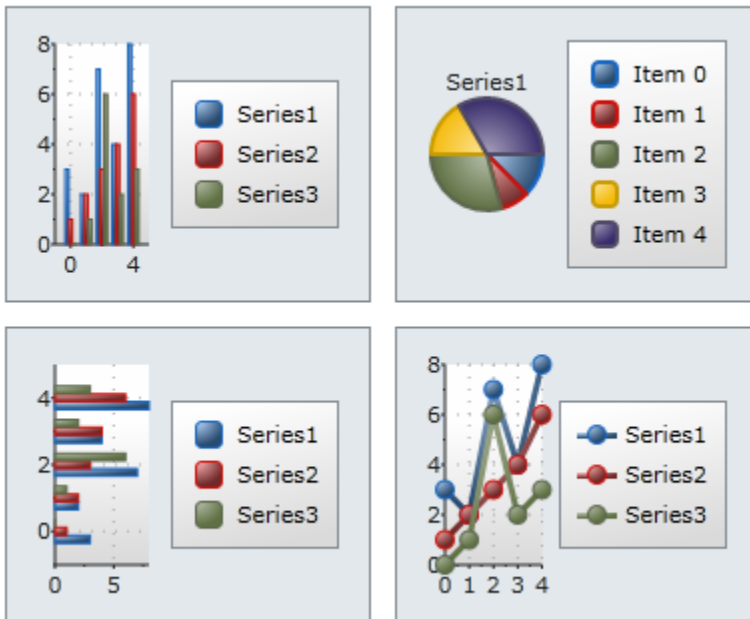
## ShinyBlue

The ShinyBlue theme is a blue-based theme similar to the Microsoft Shiny Blue theme included in the Silverlight Toolkit. The ShinyBlue theme appears similar to the following when applied to the **ComponentOne Studio for Silverlight** charting controls:



## WhistlerBlue

The WhistlerBlue theme is a blue-based theme similar to the Microsoft Whistler Blue theme, which is included in the Silverlight Toolkit. The WhistlerBlue theme appears similar to the following when applied to the **ComponentOne Studio for Silverlight** charting controls:



## Custom Themes

In addition to using one of the built-in themes, you can create your own custom theme from scratch or create a custom theme based on an existing built-in theme. See [Included XAML Files](#) (page 26) for the included files that you can base a theme on.

## Included XAML Files

Several auxiliary XAML elements are installed with **ComponentOne Studio for Silverlight**. These elements include templates and themes and are located in the **Studio for Silverlight** installation directory. You can incorporate these elements into your project to, for example, create your own theme based on the included themes.

By default, these files are located in the **generics.zip** file in the **C:\Program Files\ComponentOne\Studio for Silverlight\Help** folder. Unzip the **generics.zip** file to a folder to see all the XAML files associated with **Studio for Silverlight** controls. In the following topics the included files are listed by assembly with their location folder within the **generics.zip** file noted.

Silverlight 4.0, Silverlight 5.0, and Windows Phone XAML files are named differently. For example, the same version of the generic.xaml file would follow the following naming conventions:

| Platform        | Name               |
|-----------------|--------------------|
| Silverlight 4.0 | generic.xaml       |
| Silverlight 5.0 | generic_SL5rd.xaml |
| Windows Phone   | generic.Phone.xaml |

The following topics list the Silverlight 4.0 file names, simply add "\_SL5rd" or ".Phone" to the file name for the Silverlight 5.0 and Windows Phone files respectively.

## C1.Silverlight

The following XAML files can be used to customize items in the **C1.Silverlight** assembly:

| Element                           | Folder                | Description   |
|-----------------------------------|-----------------------|---|
| generic.xaml                      | C1.Silverlight\themes | Specifies the templates for different styles and the initial style of the controls. |
| Common.xaml                       | C1.Silverlight\themes | Specifies attributes for common elements in the controls.                           |
| C1Button.xaml                     | C1.Silverlight\themes | Specifies attributes for C1Button.  |
| C1ComboBox.xaml                   | C1.Silverlight\themes | Specifies attributes for C1ComboBox.  |
| C1DateTimePicker.xaml             | C1.Silverlight\themes | Specifies attributes for C1DateTimePicker.  |
| C1DropDown.xaml                   | C1.Silverlight\themes | Specifies attributes for C1DropDown.  |
| C1FilePicker.xaml                 | C1.Silverlight\themes | Specifies attributes for C1FilePicker.  |
| C1HeaderedContentControl.xaml     | C1.Silverlight\themes | Specifies attributes for C1HeaderedContentControl.                                  |
| C1LayoutTransformer.xaml          | C1.Silverlight\themes | Specifies attributes for C1LayoutTransformer.                                       |
| C1LoopingList.xaml                | C1.Silverlight\themes | Specifies attributes for C1LoopingList.   |
| C1Menu.xaml                       | C1.Silverlight\themes | Specifies attributes for C1Menu.  |
| C1NumericBox.xaml                 | C1.Silverlight\themes | Specifies attributes for C1NumericBox.  |
| C1ProgressBar.xaml                | C1.Silverlight\themes | Specifies attributes for C1ProgressBar.   |
| C1RangeSlider.xaml                | C1.Silverlight\themes | Specifies attributes for C1RangeSlider.   |
| C1ScrollBar.xaml                  | C1.Silverlight\themes | Specifies attributes for C1ScrollBar.   |
| C1ScrollViewer.xaml               | C1.Silverlight\themes | Specifies attributes for C1ScrollViewer.  |
| C1Separator.xaml                  | C1.Silverlight\themes | Specifies attributes for C1Separator.   |
| C1TabControl.xaml                 | C1.Silverlight\themes | Specifies attributes for C1TabControl.  |
| C1TextBoxBase.xaml                | C1.Silverlight\themes | Specifies attributes for C1TextBoxBase.   |
| C1TextEditableContentControl.xaml | C1.Silverlight\themes | Specifies attributes for C1TextEditableContentControl.                              |
| C1ToggleSwitch.xaml               | C1.Silverlight\themes | Specifies attributes for C1ToggleSwitch.  |
| C1TreeView.xaml                   | C1.Silverlight\themes | Specifies attributes for C1TreeView.  |
| C1ValidationDecorator.xaml        | C1.Silverlight\themes | Specifies attributes for C1ValidationDecorator.                                     |
| C1Window.xaml                     | C1.Silverlight\themes | Specifies attributes for C1Window.  |

## C1.Silverlight.Chart

The following XAML files can be used to customize items in the **C1.Silverlight.Chart** assembly:

| Element       | Folder                        | Description  |
|---------------|-------------------------------|--|
| generic.xaml  | C1.Silverlight.Chart\themes   | Specifies the templates for different styles and the initial style of the chart. |
| DuskBlue.xaml | C1.Silverlight.Chart\ThemesSL | Specifies the attributes for the DuskBlue theme.                                 |

|                        |                                   |   |
|------------------------|-----------------------------------|---|
| DuskGreen.xaml         | C1.Silverlight.Chart\ThemesSL     | Specifies the attributes for the DuskGreen theme.             |
| MediaPlayer.xaml       | C1.Silverlight.Chart\ThemesSL     | Specifies the attributes for the MediaPlayer theme.           |
| Office2003Blue.xaml    | C1.Silverlight.Chart\ThemesSL     | Specifies the attributes for the Office2003Blue theme.        |
| Office2003Classic.xaml | C1.Silverlight.Chart\ThemesSL     | Specifies the attributes for the Office2003Classic theme.     |
| Office2003Olive.xaml   | C1.Silverlight.Chart\ThemesSL     | Specifies the attributes for the Office2003Olive theme.       |
| Office2003Royale.xaml  | C1.Silverlight.Chart\ThemesSL     | Specifies the attributes for the Office2003Royale theme.      |
| Office2003Silver.xaml  | C1.Silverlight.Chart\ThemesSL     | Specifies the attributes for the Office2003Silver theme.      |
| Office2007Black.xaml   | C1.Silverlight.Chart\ThemesSL     | Specifies the attributes for the Office2007Black theme.       |
| Office2007Blue.xaml    | C1.Silverlight.Chart\ThemesSL     | Specifies the attributes for the Office2007Blue theme.        |
| Office2007Silver.xaml  | C1.Silverlight.Chart\ThemesSL     | Specifies the attributes for the Office2007Silver theme.      |
| Vista.xaml             | C1.Silverlight.Chart\ThemesSL     | Specifies the attributes for the Vista theme.                 |
| generic.xaml           | C1.Silverlight.Chart\Phone\themes | Specifies the templates for the Metro theme for the controls. |

## C1.Silverlight.Chart.Editor

The following XAML files can be used to customize items in the **C1.Silverlight.Chart.Editor** assembly:

| Element              | Folder                                      | Description   |
|----------------------|---|---|
| AxisEditor.xaml      | C1.Silverlight.Chart.Editor                 | Specifies the attributes for the Axis Editor.       |
| ChartEditor.xaml     | C1.Silverlight.Chart.Editor                 | Specifies the attributes for the Chart Editor.      |
| DataLabelEditor.xaml | C1.Silverlight.Chart.Editor                 | Specifies the attributes for the Data Label Editor. |
| LegendEditor.xaml    | C1.Silverlight.Chart.Editor                 | Specifies the attributes for the Legend Editor.     |
| DashesEditor.xaml    | C1.Silverlight.Chart.Editor\AuxControls     | Specifies the attributes for the Dashes Editor.     |
| PropertyEditor.xaml  | C1.Silverlight.Chart.Editor\PropertyEditors | Specifies the attributes for the Property Editor.   |

## C1.Silverlight.Chart3D

The following XAML files can be used to customize items in the **C1.Silverlight.Chart3D** assembly:

| Element      | Folder                 | Description                                      |
|--------------|------------------------|--|
| generic.xaml | C1.Silverlight.Chart3D | Specifies the templates for different styles and |

|  |          |                                 |
|--|----------|---------------------------------|
|  | D\themes | the initial style of the chart. |
|--|----------|---------------------------------|

## C1.Silverlight.DataGrid

The following XAML file can be used to customize items in the **C1.Silverlight.DataGrid** assembly:

| Element   | Folder                         | Description   |
|---|--------------------------------|---|
| generic.xaml                                    | C1.Silverlight.DataGrid\themes | Specifies the templates for different styles and the initial style of the controls. |
| Common.xaml                                     | C1.Silverlight.DataGrid\themes | Specifies attributes for common elements in the controls.                           |
| DataGridCellPresenter.xaml                      | C1.Silverlight.DataGrid\themes | Specifies attributes for common elements in the controls.                           |
| DataGridColumnHeaderPresenter.xaml              | C1.Silverlight.DataGrid\themes | Specifies attributes for the column header presenter.                               |
| DataGridDetailsPresenter.xaml                   | C1.Silverlight.DataGrid\themes | Specifies attributes for the data details presenter.                                |
| DataGridDragNDrop.xaml                          | C1.Silverlight.DataGrid\themes | Specifies attributes for grid drag-and-drop operation.                              |
| DataGridFilter.xaml                             | C1.Silverlight.DataGrid\themes | Specifies attributes for the grid's filtering.                                      |
| DataGridGroupingPresenter.xaml                  | C1.Silverlight.DataGrid\themes | Specifies attributes for the grouping presenter.                                    |
| DataGridRowHeaderPresenter.xaml                 | C1.Silverlight.DataGrid\themes | Specifies attributes for the row header presenter.                                  |
| DataGridRowPresenter.xaml                       | C1.Silverlight.DataGrid\themes | Specifies attributes for the row presenter.   |
| DataGridVerticalFreezingSeparatorPresenter.xaml | C1.Silverlight.DataGrid\themes | Specifies attributes for the freezing separator presenter.                          |

## C1.Silverlight.DataGrid.Filters

The following XAML file can be used to customize items in the **C1.Silverlight.DataGrid.Filters** assembly:

| Element      | Folder                                 | Description   |
|--------------|--|---|
| generic.xaml | C1.Silverlight.DataGrid.Filters\themes | Specifies the templates for different styles and the initial style of the controls. |

## C1.Silverlight.DataGrid.Ria

The following XAML file can be used to customize items in the **C1.Silverlight.DataGrid.Ria** assembly:

| Element      | Folder                             | Description   |
|--------------|------------------------------------|---|
| generic.xaml | C1.Silverlight.DataGrid.Ria\themes | Specifies the templates for different styles and the initial style of the controls. |

## C1.Silverlight.DataGrid.Summaries

The following XAML file can be used to customize items in the **C1.Silverlight.DataGrid.Summaries** assembly:

| Element      | Folder                                   | Description   |
|--------------|--|---|
| generic.xaml | C1.Silverlight.DataGrid.Summaries\themes | Specifies the templates for different styles and the initial style of the controls. |

## C1.Silverlight.DateTimeEditors

The following XAML file can be used to customize items in the **C1.Silverlight.DateTimeEditors** assembly:

| Element      | Folder                                | Description   |
|--------------|---------------------------------------|---|
| generic.xaml | C1.Silverlight.DateTimeEditors\themes | Specifies the templates for different styles and the initial style of the controls. |

## C1.Silverlight.Docking

The following XAML file can be used to customize items in the **C1.Silverlight.Docking** assembly:

| Element      | Folder                        | Description   |
|--------------|-------------------------------|---|
| generic.xaml | C1.Silverlight.Docking\themes | Specifies the templates for different styles and the initial style of the controls. |

## C1.Silverlight.Extended

The following XAML file can be used to customize items in the **C1.Silverlight.Extended** assembly:

| Element             | Folder                               | Description   |
|---------------------|--------------------------------------|---|
| generic.xaml        | C1.Silverlight.Extended\themes       | Specifies the templates for different styles and the initial style of the controls. |
| C1Accordion.xaml    | C1.Silverlight.Extended\themes       | Specifies attributes for C1Accordion.   |
| C1Book.xaml         | C1.Silverlight.Extended\themes       | Specifies attributes for C1Book.  |
| C1ColorPicker.xaml  | C1.Silverlight.Extended\themes       | Specifies attributes for C1ColorPicker.   |
| C1CoverFlow.xaml    | C1.Silverlight.Extended\themes       | Specifies attributes for C1CoverFlow.   |
| C1Expander.xaml     | C1.Silverlight.Extended\themes       | Specifies attributes for C1Expander.  |
| C1PropertyGrid.xaml | C1.Silverlight.Extended\themes       | Specifies attributes for C1PropertyGrid.  |
| C1Reflector.xaml    | C1.Silverlight.Extended\themes       | Specifies attributes for C1Reflector.   |
| generic.xaml        | C1.Silverlight.Extended\Phone\Themes | Specifies the templates for the Metro theme of the controls.                        |
| C1Accordion.xaml    | C1.Silverlight.Extended\Phone\Themes | Specifies attributes for the Metro theme for C1Accordion.                           |
| C1Book.xaml         | C1.Silverlight.Extended\Phone\Themes | Specifies attributes for the Metro theme for C1Book.                                |
| C1ColorPicker.xaml  | C1.Silverlight.Extended\Phone\Themes | Specifies attributes for the Metro theme for C1ColorPicker.                         |

|                     |                                      |  |
|---------------------|--------------------------------------|--|
| C1CoverFlow.xaml    | C1.Silverlight.Extended\Phone\Themes | Specifies attributes for the Metro theme for C1CoverFlow.    |
| C1Expander.xaml     | C1.Silverlight.Extended\Phone\Themes | Specifies attributes for the Metro theme for C1Expander.     |
| C1PropertyGrid.xaml | C1.Silverlight.Extended\Phone\Themes | Specifies attributes for the Metro theme for C1PropertyGrid. |
| C1Reflector.xaml    | C1.Silverlight.Extended\Phone\Themes | Specifies attributes for the Metro theme for C1Reflector.    |

## C1.Silverlight.FlexGrid

The following XAML file can be used to customize items in the **C1.Silverlight.FlexGrid** assembly:

| Element      | Folder                         | Description   |
|--------------|--------------------------------|---|
| generic.xaml | C1.Silverlight.FlexGrid\themes | Specifies the templates for different styles and the initial style of the controls. |

## C1.Silverlight.FlexGrid.Filter

The following XAML file can be used to customize items in the **C1.Silverlight.FlexGrid.Filter** assembly:

| Element      | Folder                                | Description   |
|--------------|---------------------------------------|---|
| generic.xaml | C1.Silverlight.FlexGrid.Filter\themes | Specifies the templates for different styles and the initial style of the controls. |

## C1.Silverlight.Gauge

The following XAML file can be used to customize items in the **C1.Silverlight.Gauge** assembly:

| Element      | Folder                      | Description   |
|--------------|-----------------------------|---|
| generic.xaml | C1.Silverlight.Gauge\themes | Specifies the templates for different styles and the initial style of the controls. |

## C1.Silverlight.Imaging

The following XAML file can be used to customize items in the **C1.Silverlight.Imaging** assembly:

| Element      | Folder                        | Description   |
|--------------|-------------------------------|---|
| generic.xaml | C1.Silverlight.Imaging\themes | Specifies the templates for different styles and the initial style of the controls. |

## C1.Silverlight.Legacy

The following XAML file can be used to customize items in the **C1.Silverlight.Legacy** assembly:

| Element      | Folder                       | Description   |
|--------------|------------------------------|---|
| generic.xaml | C1.Silverlight.Legacy\themes | Specifies the templates for different styles and the initial style of the controls. |

## C1.Silverlight.Maps

The following XAML file can be used to customize items in the **C1.Silverlight.Maps** assembly:

| Element            | Folder                     | Description   |
|--------------------|----------------------------|---|
| generic.xaml       | C1.Silverlight.Maps\themes | Specifies the templates for different styles and the initial style of the controls. |
| ZoomScrollBar.xaml | C1.Silverlight.Maps\themes | Specifies attributes for the zoom scroll bar.                                       |

## C1.Silverlight.MediaPlayer

The following XAML file can be used to customize items in the **C1.Silverlight.MediaPlayer** assembly:

| Element      | Folder                            | Description   |
|--------------|-----------------------------------|---|
| generic.xaml | C1.Silverlight.MediaPlayer\themes | Specifies the templates for different styles and the initial style of the controls. |

## C1.Silverlight.OrgChart

The following XAML files can be used to customize items in the **C1.Silverlight.OrgChart** assembly:

| Element      | Folder                         | Description   |
|--------------|--------------------------------|---|
| generic.xaml | C1.Silverlight.OrgChart\themes | Specifies the templates for different styles and the initial style of the controls. |

## C1.Silverlight.OutlookBar

The following XAML files can be used to customize items in the **C1.Silverlight.OutlookBar** assembly:

| Element                  | Folder                           | Description   |
|--------------------------|----------------------------------|---|
| generic.xaml             | C1.Silverlight.OutlookBar\themes | Specifies the templates for different styles and the initial style of the controls. |
| OutlookBar2007.xaml      | C1.Silverlight.OutlookBar\themes | Specifies the templates for different styles and the initial style of the controls. |
| OutlookBar2007Black.xml  | C1.Silverlight.OutlookBar\themes | Specifies the templates for different styles and the initial style of the controls. |
| OutlookBar2007Silver.xml | C1.Silverlight.OutlookBar\themes | Specifies the templates for different styles and the initial style of the controls. |
| OutlookBar2010.xaml      | C1.Silverlight.OutlookBar\themes | Specifies the templates for different styles and the initial style of the controls. |
| OutlookBar2010Black.xml  | C1.Silverlight.OutlookBar\themes | Specifies the templates for different styles and the initial style of the controls. |
| OutlookBar2010Silver.xml | C1.Silverlight.OutlookBar\themes | Specifies the templates for different styles and the initial style of the controls. |

## C1.Silverlight.PdfViewer

The following XAML file can be used to customize items in the **C1.Silverlight.PdfViewer** assembly:

| Element      | Folder                          | Description   |
|--------------|---------------------------------|---|
| generic.xaml | C1.Silverlight.PdfViewer\themes | Specifies the templates for different styles and the initial style of the controls. |

## C1.Silverlight.ReportViewer

The following XAML file can be used to customize items in the **C1.Silverlight.ReportViewer** assembly:

| Element      | Folder                             | Description   |
|--------------|------------------------------------|---|
| generic.xaml | C1.Silverlight.ReportViewer\themes | Specifies the templates for different styles and the initial style of the controls. |

## C1.Silverlight.RichTextBox

The following XAML file can be used to customize items in the **C1.Silverlight.RichTextBox** assembly:

| Element      | Folder                            | Description   |
|--------------|-----------------------------------|---|
| generic.xaml | C1.Silverlight.RichTextBox\themes | Specifies the templates for different styles and the initial style of the controls. |

## C1.Silverlight.RichTextBox.Toolbar

The following XAML file can be used to customize items in the **C1.Silverlight.RichTextBox.Toolbar** assembly:

| Element      | Folder                                    | Description   |
|--------------|---|---|
| generic.xaml | C1.Silverlight.RichTextBox.Toolbar\themes | Specifies the templates for different styles and the initial style of the controls. |

## C1.Silverlight.Schedule

The following XAML files can be used to customize items in the **C1.Silverlight.Schedule** assembly:

| Element                                 | Folder                          | Description  |
|---|---------------------------------|--|
| EditAppointmentControl.Silverlight.xaml | C1.Silverlight.Schedule\Dialogs | Specifies the attributes for editing appointments.           |
| EditCollectionControl.xaml              | C1.Silverlight.Schedule\Dialogs | Specifies the attributes for editing collections.            |
| EditRecurrenceControl.Silverlight.xaml  | C1.Silverlight.Schedule\Dialogs | Specifies the attributes for editing appointment recurrence. |
| RecChoiceControl.Silverlight.xaml       | C1.Silverlight.Schedule\Dialogs | Specifies the attributes for choosing recurrence.            |
| SelectFromListScene.Silverlight.xaml    | C1.Silverlight.Schedule\Dialogs | Specifies the attributes for resources from lists.           |
| SelectFromListScene.WPF.xaml            | C1.Silverlight.Schedule\Dialogs | Specifies the attributes for resources from lists.           |
| ShowRemindersControl.Silverlight.xaml   | C1.Silverlight.Schedule\Dialogs | Specifies the attributes for schedule reminders.             |

|                                   |                                  |   |
|-----------------------------------|----------------------------------|---|
| generic.xaml                      | C1.Silverlight.Scheduler\Dialogs | Specifies the templates for different styles and the initial style of the controls. |
| Auxiliary.xaml                    | C1.Silverlight.Scheduler\themes  | Specifies attributes for auxiliary elements of the control.                         |
| C1Calendar.xaml                   | C1.Silverlight.Scheduler\themes  | Specifies attributes for the C1Calendar.  |
| C1SchedulerParts.xaml             | C1.Silverlight.Scheduler\themes  | Specifies attributes for parts of the scheduler.                                    |
| Common.xaml                       | C1.Silverlight.Scheduler\themes  | Specifies attributes for common elements of the scheduler.                          |
| generic.xaml                      | C1.Silverlight.Scheduler\themes  | Specifies the templates for different styles and the initial style of the controls. |
| IntervalAppointmentPresenter.xaml | C1.Silverlight.Scheduler\themes  | Specifies attributes for the interval appointment presenter.                        |

### C1.Silverlight.SpellChecker

The following XAML file can be used to customize items in the **C1.Silverlight.SpellChecker** assembly:

| Element            | Folder                      | Description  |
|--------------------|-----------------------------|--|
| C1SpellDialog.xaml | C1.Silverlight.SpellChecker | Specifies the attributes for the Spell Checker Dialog Box. |

### C1.Silverlight.Theming.BureauBlack

The following XAML files can be used to customize items in the **C1.Silverlight.BureauBlack** assembly:

| Element  | Folder                             | Description   |
|--|------------------------------------|---|
| BureauBlack.xaml                                 | C1.Silverlight.Theming.BureauBlack | Specifies resources and styling elements for each ComponentOne Silverlight control. |
| System.Windows.Controls.Theming.BureauBlack.xaml | C1.Silverlight.Theming.BureauBlack | Specifies the standard Microsoft BureauBlack resources and styling elements.        |
| Theme.xaml                                       | C1.Silverlight.Theming.BureauBlack | Specifies the standard resources and styling elements.                              |

### C1.Silverlight.Theming.Cosmopolitan

The following XAML files can be used to customize items in the **C1.Silverlight.Cosmopolitan** assembly:

| Element                                   | Folder                              | Description   |
|---|-------------------------------------|---|
| Cosmopolitan.xaml/Cosmopolitan_SL5rd.xaml | C1.Silverlight.Theming.Cosmopolitan | Specifies resources and styling elements for each ComponentOne Silverlight control. |
| Merged.xaml                               | C1.Silverlight.Theming.Cosmopolitan | Specifies the resources for each ComponentOne WPF control.                          |
| Theme.xaml/Theme_SL5rd.xaml               | C1.Silverlight.Theming.Cosmopolitan | Specifies the standard resources and styling elements.                              |

## C1.Silverlight.Theming.ExpressionDark

The following XAML files can be used to customize items in the **C1.Silverlight.ExpressionDark** assembly:

| Element   | Folder                                | Description   |
|---|---------------------------------------|---|
| ExpressionDark.xaml                                 | C1.Silverlight.Theming.ExpressionDark | Specifies resources and styling elements for each ComponentOne Silverlight control. |
| System.Windows.Controls.Theming.ExpressionDark.xaml | C1.Silverlight.Theming.ExpressionDark | Specifies the standard Microsoft ExpressionDark resources and styling elements.     |
| Theme.xaml  | C1.Silverlight.Theming.ExpressionDark | Specifies the standard resources and styling elements.                              |

## C1.Silverlight.Theming.ExpressionLight

The following XAML files can be used to customize items in the **C1.Silverlight.ExpressionLight** assembly:

| Element  | Folder                                 | Description   |
|--|--|---|
| ExpressionLight.xaml                                 | C1.Silverlight.Theming.ExpressionLight | Specifies resources and styling elements for each ComponentOne Silverlight control. |
| System.Windows.Controls.Theming.ExpressionLight.xaml | C1.Silverlight.Theming.ExpressionLight | Specifies the standard Microsoft ExpressionLight resources and styling elements.    |
| Theme.xaml   | C1.Silverlight.Theming.ExpressionLight | Specifies the standard resources and styling elements.                              |

## C1.Silverlight.Theming.Office2007

The following XAML files can be used to customize items in the **C1.Silverlight.Office2007** assembly:

| Element               | Folder                            | Description  |
|-----------------------|-----------------------------------|--|
| Office2007.xaml       | C1.Silverlight.Theming.Office2007 | Specifies the standard resources and styling elements. |
| Office2007Black.xaml  | C1.Silverlight.Theming.Office2007 | Specifies the standard resources and styling elements. |
| Office2007Silver.xaml | C1.Silverlight.Theming.Office2007 | Specifies the standard resources and styling elements. |
| Theme.xaml            | C1.Silverlight.Theming.Office2007 | Specifies the standard resources and styling elements. |

## C1.Silverlight.Theming.Office2010

The following XAML files can be used to customize items in the **C1.Silverlight.Office2010** assembly:

| Element               | Folder                            | Description  |
|-----------------------|-----------------------------------|--|
| Office2010.xaml       | C1.Silverlight.Theming.Office2010 | Specifies the standard resources and styling elements. |
| Office2010Black.xaml  | C1.Silverlight.Theming.Office2010 | Specifies the standard resources and styling elements. |
| Office2010Silver.xaml | C1.Silverlight.Theming            | Specifies the standard resources and styling           |

|            |                                       |  |
|------------|---------------------------------------|--|
|            | g.Office2010                          | elements.  |
| Theme.xaml | C1.Silverlight.Themin<br>g.Office2010 | Specifies the standard resources and styling elements. |

### C1.Silverlight.Theming.RainierOrange

The following XAML files can be used to customize items in the **C1.Silverlight.RainierOrange** assembly:

| Element            | Folder                                   | Description   |
|--------------------|--|---|
| RainierOrange.xaml | C1.Silverlight.Themin<br>g.RainierOrange | Specifies resources and styling elements for each ComponentOne Silverlight control. |
| Theme.xaml         | C1.Silverlight.Themin<br>g.RainierOrange | Specifies the standard resources and styling elements.                              |

### C1.Silverlight.Theming.ShinyBlue

The following XAML files can be used to customize items in the **C1.Silverlight.ShinyBlue** assembly:

| Element        | Folder                               | Description   |
|----------------|--------------------------------------|---|
| ShinyBlue.xaml | C1.Silverlight.Themin<br>g.ShinyBlue | Specifies resources and styling elements for each ComponentOne Silverlight control. |
| Theme.xaml     | C1.Silverlight.Themin<br>g.ShinyBlue | Specifies the standard resources and styling elements.                              |

### C1.Silverlight.Theming.WhistlerBlue

The following XAML files can be used to customize items in the **C1.Silverlight.WhistlerBlue** assembly:

| Element           | Folder                                  | Description   |
|-------------------|---|---|
| WhistlerBlue.xaml | C1.Silverlight.Themin<br>g.WhistlerBlue | Specifies resources and styling elements for each ComponentOne Silverlight control. |
| Theme.xaml        | C1.Silverlight.Themin<br>g.WhistlerBlue | Specifies the standard resources and styling elements.                              |

### C1.Silverlight.TileView

The following XAML files can be used to customize items in the **C1.Silverlight.TileView** assembly:

| Element      | Folder                             | Description   |
|--------------|------------------------------------|---|
| generic.xaml | C1.Silverlight.TileVie<br>w\themes | Specifies the templates for different styles and the initial style of the controls. |

### C1.Silverlight.Toolbar

The following XAML files can be used to customize items in the **C1.Silverlight.Toolbar** assembly:

| Element      | Folder                            | Description   |
|--------------|-----------------------------------|---|
| generic.xaml | C1.Silverlight.Toolbar<br>\themes | Specifies the templates for different styles and the initial style of the controls. |

|                   |                                  |  |
|-------------------|----------------------------------|--|
| C1ToolBarTab.xaml | C1.Silverlight.Toolbar<br>themes | Specifies the attributes for the C1ToolBarTab. |
|-------------------|----------------------------------|--|

## Implicit and Explicit Styles

The following topic detail using implicit and explicit styles and using the **ImplicitStyleManager** which is included in the Silverlight Toolkit. For more information about the Silverlight Toolkit, see [CodePlex](#).

### Implicit Styles

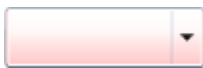
If you're familiar with WPF (Windows Presentation Foundation) you may be used to setting styles implicitly so the application has a uniform appearance – for example, you're used to setting the style for all instances of a particular control in the application's resources. Unfortunately Silverlight does not support implicit styles in the same way that WPF does and you would normally have to indicate the style to use in each instance of the control. This can be tedious to do if you have several controls on a page and that's where the **ImplicitStyleManager** comes in handy. The **ImplicitStyleManager** class is located in the Microsoft.Windows.Controls.Theming namespace (in the Microsoft.Windows.Controls assembly).

### WPF and Silverlight Styling

In WPF, you can set styles implicitly. When you set styles implicitly all instances of a particular type can be styled at once. For example, the WPF **C1DropDown** control might be styled with the following markup:

```
<Grid>
  <Grid.Resources>
    <Style TargetType="{x:Type c1:C1DropDown}">
      <Setter Property="Background" Value="Red" />
    </Style>
  </Grid.Resources>
  <c1:C1DropDown Height="30" HorizontalAlignment="Center"
    Name="C1DropDown1" VerticalAlignment="Center" Width="100" />
</Grid>
```

This would set the background of the control to be the color red as in the following image:



All **C1DropDown** controls in the grid would also appear red; **C1DropDown** controls outside of the Grid would not appear red. This is what is meant by implicit styles – the style is assigned to all controls of a particular type. Inherited controls would also inherit the style.

Silverlight, however, does not support implicit styles. In Silverlight you could add the style to the Grid's resources similarly:

```
<Grid.Resources>
  <Style x:Key="DropDownStyle" TargetType="c1:C1DropDown">
    <Setter Property="Background" Value="Red" />
  </Style>
</Grid.Resources>
```

But the Silverlight **C1DropDown** control would not be styled unless the style was explicitly set, as in the following example:

```
<c1:C1DropDown Height="30" HorizontalAlignment="Center" Name="C1DropDown1"
  VerticalAlignment="Center" Width="100" Style="{StaticResource
  DropDownStyle}"/>
```

While this is easy enough to set on one control, if you have several controls it can be tedious to set the style on each one. That's where the **ImplicitStyleManager** comes in. See [Using the ImplicitStyleManager](#) (page 38) for more information.

## Using the ImplicitStyleManager

The **ImplicitStyleManager** lets you set styles implicitly in Silverlight as you might in WPF. You can find the **ImplicitStyleManger** in the **System.Windows.Controls.Theming.Toolkit.dll** assembly installed with the Silverlight Toolkit.

To use the **ImplicitStyleManager** add a reference in your project to the **System.Windows.Controls.Theming.Toolkit.dll** assembly and add its namespace to the initial **UserControl** tag as in the following markup:

```
<UserControl
  xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
  xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml" xmlns:c1="clr-
  namespace:C1.Silverlight;assembly=C1.Silverlight" xmlns:theming="clr-
  namespace:System.Windows.Controls.Theming;assembly=System.Windows.Controls.Th
  eming.Toolkit" x:Class="C1Theming.MainPage" Width="640" Height="480">
```

Once you've added the reference and namespace you can use the **ImplicitStyleManager** in your application. For example, in the following markup a style is added and implicitly implemented:

```
<Grid x:Name="LayoutRoot" Background="White"
  theming:ImplicitStyleManager.ApplyMode="OneTime">
  <Grid.Resources>
    <Style TargetType="c1:C1DropDown">
      <Setter Property="Background" Value="Red" />
    </Style>
  </Grid.Resources>
  <StackPanel HorizontalAlignment="Center" VerticalAlignment="Center">
    <c1:C1DropDown Margin="5" Content="C1DropDown" Height="30"
    Width="100"/>
  </StackPanel>
</Grid>
```

## Applying Themes to Controls

You can easily customize your application, by applying one of the built-in themes to your ComponentOne Silverlight control. Each of the built-in themes is based on a Silverlight Toolkit theme. For information about each of the built-in themes, see [Available Themes](#) (page 21). In this example, you'll add the RainierOrange theme to the **C1DropDown** control on a page.

To apply the theme, complete the following steps:

1. In Visual Studio, select **File | New Project**.

- In the **New Project** dialog box, select the language in the left pane and in the right-pane select **Silverlight Application**. Enter a **Name** and **Location** for your project and click **OK**.
- In the **New Silverlight Application** dialog box, leave the default settings and click **OK**.

A new application will be created and should open with the **MainPage.xaml** file displayed in XAML view.

- Place the mouse cursor between the `<Grid>` and `</Grid>` tags in XAML view.

You will add the theme and control to the Grid in the next steps.

- Navigate to the Visual Studio Toolbox and double-click on the **C1ThemeRainierOrange** icon to declare the theme. The theme's namespace will be added to the page and the theme's tags will be added to the Grid in XAML view. The markup will appear similar to the following:

```
<UserControl xmlns:my="clr-
namespace:C1.Silverlight.Theming.RainierOrange;assembly=C1.Silverlight.
Theming.RainierOrange" x:Class="C1Silverlight.MainPage"
xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
xmlns:d="http://schemas.microsoft.com/expression/blend/2008"
xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"
mc:Ignorable="d" d:DesignWidth="640" d:DesignHeight="480">
  <Grid x:Name="LayoutRoot">
    <my:C1ThemeRainierOrange></my:C1ThemeRainierOrange>
  </Grid>
</UserControl>
```

Any controls that you add within the theme's tags will now be themed.

- Place your cursor between the `<my:C1ThemeRainierOrange>` and `</my:C1ThemeRainierOrange>` tags.
- In the Toolbox, double-click the **C1DropDown** icon to add the control to the project. The `C1.Silverlight` namespace will be added to the page and the control's tags will be added within the theme's tags in XAML view. The markup will appear similar to the following:

```
<UserControl xmlns:c1="clr-
namespace:C1.Silverlight;assembly=C1.Silverlight"
xmlns:my="clr-
namespace:C1.Silverlight.Theming.RainierOrange;assembly=C1.Silverlight.
Theming.RainierOrange" x:Class="C1Silverlight.MainPage"
xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
xmlns:d="http://schemas.microsoft.com/expression/blend/2008"
xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"
mc:Ignorable="d" d:DesignWidth="640" d:DesignHeight="480">
  <Grid x:Name="LayoutRoot">
    <my:C1ThemeRainierOrange>
      <c1:C1DropDown Width="100" Height="30"></c1:C1DropDown>
    </my:C1ThemeRainierOrange>
  </Grid>
</UserControl>
```

## What You've Accomplished

Run your project and observe that the **C1DropDown** control now appears in the **RainierOrange** theme. Note that you can only set the **Content** property on the theme once, so to theme multiple controls using this method you will need to add a panel, for example a **Grid** or **StackPanel**, within the theme and then add multiple controls within the panel.

You can also use the **ImplicitStyleManager** to theme all controls of a particular type. For more information, see [Using the ImplicitStyleManager](#) (page 38).

## Applying Themes to an Application

The following topic details one method of applying a theme application-wide in Visual Studio. In this topic you'll add a class to your application that initializes a built-in theme. You'll then apply the theme to the MainPage of your application.

To apply the theme, complete the following steps:

1. In Visual Studio, select **File | New Project**.
2. In the **New Project** dialog box, select the language in the left pane and in the right-pane select **Silverlight Application**. Enter a **Name** and **Location** for your project and click **OK**.
3. In the **New Silverlight Application** dialog box, leave the default settings and click **OK**.

A new application will be created and should open with the **MainPage.xaml** file displayed in XAML view.

4. In the Solution Explorer, right-click the project and choose **Add Reference**.
5. In the **Add Reference** dialog box choose the **C1.Silverlight.Theming** and **C1.Silverlight.Theming.RainierOrange** assemblies and click **OK**.
6. In the Solution Explorer, right-click the project and select **Add | New Item**.
7. In the **Add New Item** dialog box, choose **Class** from the templates list, name the class "MyThemes", and click the **Add** button to create and a new class. The newly created **MyThemes** class will open.
8. Add the following import statements to the top of the class:

- Visual Basic

```
Imports C1.Silverlight.Theming
Imports C1.Silverlight.Theming.RainierOrange
```

- C#

```
using C1.Silverlight.Theming;
using C1.Silverlight.RainierOrange;
```

9. Add code to the class so it appears like the following:

- Visual Basic

```
Public Class MyThemes
    Private _myTheme As C1Theme = Nothing
    Public ReadOnly Property MyTheme() As C1Theme
        Get
            If _myTheme Is Nothing Then
                _myTheme = New C1ThemeRainierOrange()
            End If
            Return _myTheme
        End Get
    End Property
End Class
```

- C#

```
public class MyThemes
{
    private static C1Theme _myTheme = null;
    public static C1Theme MyTheme
    {
        get
```

```

        {
            if (_myTheme == null)
                _myTheme = new C1ThemeRainierOrange();
            return _myTheme;
        }
    }
}

```

10. In the Solution Explorer, double-click the **App.xaml.vb** or **App.xaml.cs** file.
11. Add the following import statement to the top of the file, where *ProjectName* is the name of your application:
  - Visual Basic
 

```
Imports ProjectName
```
  - C#
 

```
using ProjectName;
```
12. Add code to the **Application\_Startup** event of the **App.xaml.vb** or **App.xaml.cs** file so it appears like the following:

- Visual Basic
 

```

Private Sub Application_Startup(ByVal o As Object, ByVal e As
StartupEventArgs) Handles Me.Startup
    Dim MyMainPage As New MainPage()
    Dim themes As New MyThemes
    themes.MyTheme.Apply(MyMainPage)
    Me.RootVisual = MyMainPage
End Sub

```

- C#
 

```

private void Application_Startup(object sender, StartupEventArgs e)
{
    MainPage MyMainPage = new MainPage();
    MyThemes.MyTheme.Apply(MyMainPage);
    this.RootVisual = MyMainPage;
}

```

Now any control you add to the **MainPage.xaml** file will automatically be themed.

13. Return to the **MainPage.xaml** file and place the mouse cursor between the `<Grid>` and `</Grid>` tags in XAML view.
14. In the Toolbox, double-click the **C1DropDown** icon to add the control to the project.
15. Update the control's markup so it appears like the following:

```
<c1:C1DropDown Width="100" Height="30"></c1:C1DropDown>
```

## What You've Accomplished

Run your project and observe that the **C1DropDown** control now appears in the **RainierOrange** theme. To change the theme chosen, now all you would need to do is change the theme in the **MyThemes** class.

For example, to change to the **ExpressionDark** theme:

1. Add a reference to the **C1.Theming.Silverlight.ExpressionDark.dll** assembly.
2. Open the **MyThemes** class in your project and add the following import statements to the top of the class:
  - Visual Basic
 

```
Imports C1.Silverlight.Theming.ExpressionDark
```
  - C#

```
using Cl.Silverlight.Theming.ExpressionDark;
```

3. Update code in the class so it appears like the following:

- Visual Basic

```
Public Class MyThemes
    Private _myTheme As ClTheme = Nothing
    Public ReadOnly Property MyTheme() As ClTheme
        Get
            If _myTheme Is Nothing Then
                _myTheme = New ClThemeExpressionDark()
            End If
            Return _myTheme
        End Get
    End Property
End Class
```

- C#

```
public class MyThemes
{
    private static ClTheme _myTheme = null;
    public static ClTheme MyTheme
    {
        get
        {
            if (_myTheme == null)
                _myTheme = new ClThemeExpressionDark();
            return _myTheme;
        }
    }
}
```

Note that the above steps apply the theme to the **MainPage.xaml** file. To apply the theme to additional pages, you would need to add the following code to each page:

- Visual Basic

```
Dim themes As New MyThemes
themes.MyTheme.Apply(MyMainPage)
```

- C#

```
MyThemes.MyTheme.Apply(LayoutRoot);
```

The theme will then be applied to the page. So, you only have to change one line of code to the class to change the theme, and you only have to add one line of code to each page to apply the theme.

## ComponentOne ClearStyle Technology

ComponentOne ClearStyle™ technology is a new, quick and easy approach to providing Silverlight and WPF control styling. ClearStyle allows you to create a custom style for a control without having to deal with the hassle of XAML templates and style resources.

Currently, to add a theme to all standard Silverlight controls, you must create a style resource template. In Microsoft Visual Studio this process can be difficult; this is why Microsoft introduced Expression Blend to make the task a bit easier. Having to jump between two environments can be a bit challenging to developers who are not familiar with Blend or do not have the time to learn it. You could hire a designer, but that can complicate things when your designer and your developers are sharing XAML files.

That's where ClearStyle comes in. With ClearStyle the styling capabilities are brought to you in Visual Studio in the most intuitive manner possible. In most situations you just want to make simple styling changes to the controls in your application so this process should be simple. For example, if you just want to change the row color of your

data grid this should be as simple as setting one property. You shouldn't have to create a full and complicated-looking template just to simply change a few colors.

## How ClearStyle Works

Each key piece of the control's style is surfaced as a simple color property. This leads to a unique set of style properties for each control. For example, a **Gauge** has **PointerFill** and **PointerStroke** properties, whereas a **DataGrid** has **SelectedBrush** and **MouseOverBrush** for rows.

Let's say you have a control on your form that does not support ClearStyle. You can take the XAML resource created by ClearStyle and use it to help mold other controls on your form to match (such as grabbing exact colors). Or let's say you'd like to override part of a style set with ClearStyle (such as your own custom scrollbar). This is also possible because ClearStyle can be extended and you can override the style where desired.

ClearStyle is intended to be a solution to quick and easy style modification but you're still free to do it the old fashioned way with ComponentOne's controls to get the exact style needed. ClearStyle does not interfere with those less common situations where a full custom design is required.

## ClearStyle Properties

With each release, ComponentOne will be adding ClearStyle functionality to more controls. Currently several Silverlight and WPF controls support ClearStyle. The following table lists all of the ClearStyle-supported Silverlight controls as well as the ClearStyle properties that each supports.

| Property                 | Supported Controls  |
|--------------------------|---|
| AlternatingBackground    | C1Scheduler   |
| AppointmentForeground    | C1Scheduler   |
| AlternatingRowBackground | C1DataGrid  |
| AlternatingRowForeground | C1DataGrid  |
| Background               | C1Accordion, C1AccordionItem, C1ColorPicker, C1ComboBox, C1ComboBoxItem, C1ContextMenu, C1CoverFlow, C1DataGrid, C1DateTimePicker, C1Docking, C1DropDown, C1Expander, C1ExpanderButton, C1FilePicker, C1HeaderedContentControl, C1Map, C1MediaPlayer, C1Menu, C1MenuItem, C1NumericBox, C1Window, C1RangeSlider, C1PropertyGrid, C1Scheduler, C1TabControl, C1TabItem, C1TextBoxBase, C1TimeEditor, C1ToolBar, C1ToolBarGroup, C1ToolBarStrip, C1ToolBarStripItem, C1TreeView, C1TreeViewItem, C1Window |
| ButtonBackground         | C1ComboBox, C1CoverFlow, C1DropDown, C1FilePicker, C1NumericBox, C1TimeEditor, C1ToolBarStrip, C1Window   |
| ButtonForeground         | C1ComboBox, C1CoverFlow, C1DropDown, C1FilePicker, C1NumericBox, C1TimeEditor, C1ToolBarStrip, C1Window   |
| CaretBrush               | C1ColorPicker, C1ComboBox, C1DateTimePicker, C1NumericBox, C1TextBoxBase, C1TimeEditor  |
| CategoryBackground       | C1PropertyGrid  |
| CategoryForeground       | C1PropertyGrid  |
| ControlBackground        | C1Scheduler   |
| ControlForeground        | C1Scheduler   |
| ExpandedBackground       | C1AccordionItem, C1Expander, C1ExpanderButton,  |
| FocusBrush               | C1ColorPicker, C1ComboBox, C1DataGrid, C1DateTimePicker, C1DropDown, C1Expander, C1ExpanderButton, C1FilePicker, C1MediaPlayer, C1NumericBox, C1Window, C1RangeSlider,  |

|                          |   |
|--------------------------|---|
|                          | C1TextBoxBase, C1TimeEditor, C1Toolbar, C1ToolbarGroup, C1ToolbarStrip, C1ToolbarStripItem  |
| Header                   | C1Accordion, C1AccordionItem, C1Expander, C1HeaderedContentControl, C1Window  |
| HighlightedBackground    | C1ContextMenu, C1Menu, C1MenuList, C1MenuItem   |
| HorizontalGridLinesBrush | C1DataGrid  |
| MouseOverBrush           | C1Accordion, C1AccordionItem, C1ColorPicker, C1ComboBox, C1ComboBoxItem, C1CoverFlow, C1DataGrid, C1DateTimePicker, C1Docking, C1DropDown, C1Expander, C1ExpanderButton, C1FilePicker, C1Map, C1MediaPlayer, C1NumericBox, C1RangeSlider, C1PropertyGrid, C1TabControl, C1TabItem, C1TextBoxBase, C1TimeEditor, C1Toolbar, C1ToolbarGroup, C1ToolbarStrip, C1ToolbarStripItem, C1TreeView, C1TreeViewItem, C1Window |
| OpenedBackground         | C1ContextMenu, C1Menu, C1MenuList, C1MenuItem   |
| PressedBrush             | C1ColorPicker, C1ComboBox, C1CoverFlow, C1DataGrid, C1DateTimePicker, C1DropDown, C1ExpanderButton, C1FilePicker, C1Map, C1MediaPlayer, C1NumericBox, C1PropertyGrid, C1RangeSlider, C1TextBoxBase, C1TimeEditor, C1Toolbar, C1ToolbarGroup, C1ToolbarStrip, C1ToolbarStripItem, C1Window   |
| RowBackground            | C1DataGrid  |
| RowForeground            | C1DataGrid  |
| SelectedBackground       | C1ComboBox, C1ComboBoxItem, C1DataGrid, C1Scheduler, C1TabControl, C1TabItem, C1TreeView, C1TreeViewItem,   |
| SelectionBackground      | C1ColorPicker, C1ComboBox, C1DateTimePicker, C1FilePicker, C1NumericBox, C1TextBoxBase, C1TimeEditor  |
| SelectionForeground      | C1ColorPicker, C1ComboBox, C1DateTimePicker, C1FilePicker, C1NumericBox, C1TextBoxBase, C1TimeEditor  |
| TabItemBackground,       | C1Docking   |
| TabStripBackground       | C1Docking, C1TabControl   |
| TabStripForeground       | C1Docking, C1TabControl   |
| TodayBackground          | C1Scheduler   |

# DockControl for Silverlight

Handle multiple windows in your Silverlight application with **ComponentOne DockControl™ for Silverlight**. Similar to the docking system in Microsoft Visual Studio 2008, **DockControl** delivers dockable, floating, and tabbed windows. You can also auto-hide sections and easily style the **DockControl**.

## DockControl for Silverlight Features

- **Docking Diamond**

By default, **DockControl** uses Microsoft docking diamonds like Visual Studio. See [Docking Diamond and Zones](#) (page 52) for more information.
- **Natural Docking**

Customize **DockControl** to provide natural docking allowing you to dock and float multiple windows within your application. Dock indicators appear when you drag windows over dock zones signaling to dock.
- **Floating Windows**

Allow a window to float above your application in separate windows.
- **Tabbed Windows**

Documents opened in instances of the editor are automatically arranged on tabbed panes.
- **Auto Hide**

See more of your code at one time with auto hide using the pushpin. This allows you to minimize not-in-use tool windows along the edges of the IDE.
- **Rich Programmatic API**

For complete flexibility and to restrict behavior, the docking state of any dockable window can be controlled programmatically through the rich programmatic API.
- **Silverlight Toolkit Themes Support**

Add style to your UI with built-in support for the most popular Microsoft Silverlight Toolkit themes, including ExpressionDark, ExpressionLight, WhistlerBlue, RainerOrange, ShinyBlue, and BureauBlack. See [DockControl Theming](#) (page 54) for an example of each.
- **Supports ClearStyle Technology**

**DockControl for Silverlight** supports ComponentOne's ClearStyle technology, which allows you to easily change control colors without having to change control templates. By setting a few color properties, you can quickly style the **C1DockControl** elements. See [DockControl ClearStyle Properties](#) (page 56) for more information on the ComponentOne ClearStyle technology.

## DockControl for Silverlight Quick Start

The following quick start guide is intended to get you up and running with **DockControl for Silverlight**. In this quick start, you'll start in Visual Studio to create a new project, add a **C1DockControl** to your application, and then add a **C1DockTabControl** with **C1DockTabItems**.

## Step 1 of 3: Creating a Silverlight Application

In this step you'll create a Silverlight application in Visual Studio 2008 using **ComponentOne DockControl for Silverlight**.

To set up your project and add a C1DockControl control to your application, complete the following steps:

1. In Visual Studio 2008, select **File | New | Project**.
2. In the **New Project** dialog box, select a language in the left pane (in this example, C# is used), and in the templates list select **Silverlight Application**.
3. Enter a **Name** for your project and click **OK**. The **New Silverlight Application** dialog box will appear.
4. Uncheck the **Host the Silverlight application in a new Web site** box, if necessary, and click **OK**. The **MainPage.xaml** file should open.
5. In the XAML window of the project, place the cursor between the <Grid> and </Grid> tags and click once. Note that you cannot currently add Silverlight controls directly to the design area in Visual Studio, so you must add them to the XAML window as directed in the next step.
6. Navigate to the Toolbox and double-click the C1DockControl icon to add the C1DockControl to **MainPage.xaml**. The XAML markup will now look similar to the following:

```
<UserControl
xmlns:c1="http://schemas.componentone.com/winfx/2006/xaml"
  x:Class="C1DockControlQuickStart.MainPage"
  xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
  xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
  xmlns:d="http://schemas.microsoft.com/expression/blend/2008"
  xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"
  mc:Ignorable="d" d:DesignWidth="640" d:DesignHeight="480">
  <Grid x:Name="LayoutRoot">
  <c1:C1DockControl></c1:C1DockControl>
  </Grid>
</UserControl>
```

Note that the C1.Silverlight.Docking namespace and <c1:C1DockControl></c1:C1DockControl> tags have been added to the project.

In the next step, you will add a C1DockTabControl with C1DockTabItems.

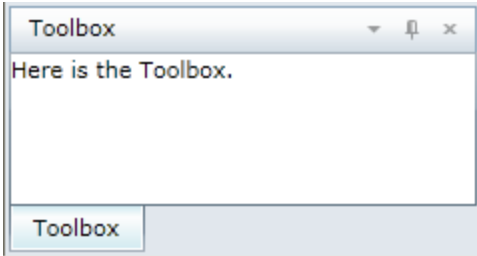
## Step 2 of 3: Adding a C1DockTabControl with C1DockTabItems

Next we are going to add a C1DockTabControl with C1DockTabItems.

1. In the XAML markup, place your cursor between the <c1:C1DockControl></c1:C1DockControl> tags and press ENTER.
2. Add a C1DockTabControl within these tags using the following XAML markup:

```
<c1:C1DockTabControl Dock="Left">
  <c1:C1DockTabItem Header="Toolbox">
    <TextBlock Text="Here is the Toolbox." />
  </c1:C1DockTabItem>
</c1:C1DockTabControl>
```

This XAML also includes a C1DockTabItem labeled **Toolbox**. If you run the application now, the C1DockTabControl will look similar to the following:



Let's add another C1DockTabItem and C1DockTabControl to show how you can select tabs and dock or float the C1DockTabControl when you run the application.

3. Add XAML for another C1DockTabItem and a C1DockTabControl after the closing </c1:C1DockTabItem> tag, so the full XAML for C1DockControl will look like the following:

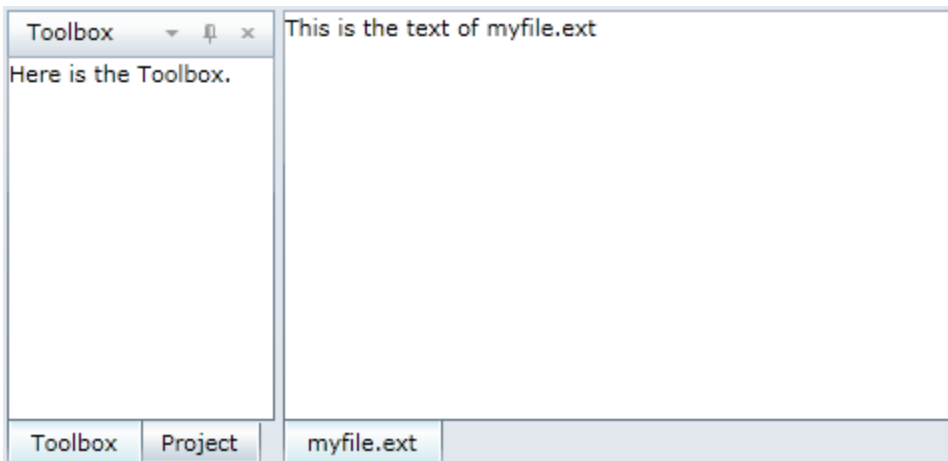
```
<c1:C1DockControl>
  <c1:C1DockTabControl Dock="Left">
    <c1:C1DockTabItem Header="Toolbox">
      <TextBlock Text="Here is the Toolbox." />
    </c1:C1DockTabItem>
    <c1:C1DockTabItem Header="Project">
      <TextBlock Text="Tree of files" />
    </c1:C1DockTabItem>
  </c1:C1DockTabControl>
  <c1:C1DockTabControl DockWidth="500" ShowHeader="False">
    <c1:C1DockTabItem Header="myfile.ext">
      <TextBlock Text="This is the text of myfile.ext" />
    </c1:C1DockTabItem>
  </c1:C1DockTabControl>
</c1:C1DockControl>
```

In the next step you will run the application and change the docking mode for windows.

### Step 3 of 3: Running the Application

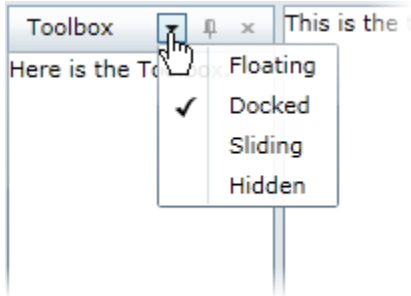
Now that you've created a Silverlight application with a C1DockControl and tab items, you're ready to run the application. Complete the following steps:

1. From the **Debug** menu, select **Start Debugging** to view how your application will appear at run time. Your application will look similar to the following:




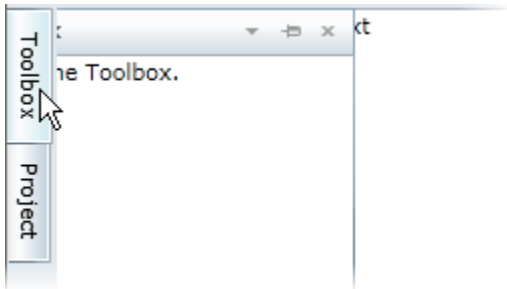
The **Toolbox** is docked on the left as we specified in the XAML.


2. Click the drop-down arrow in the **Toolbox** header and select an option to float, slide, or hide the `C1DockControl`.



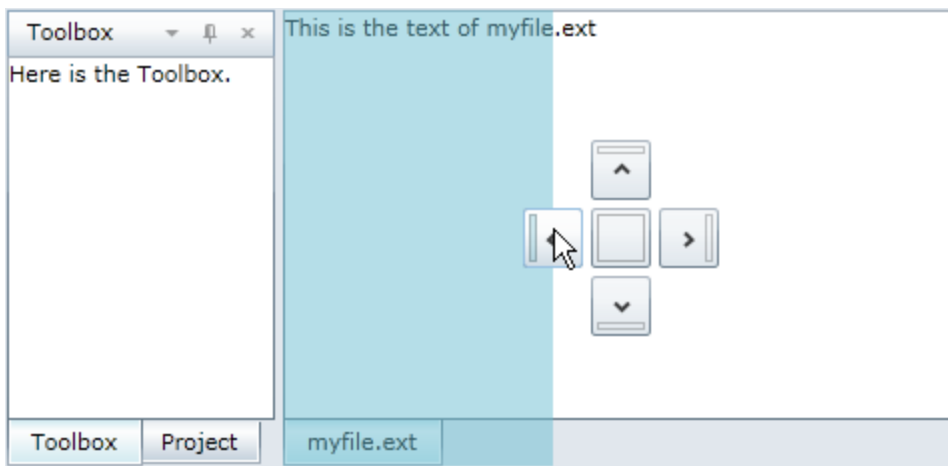
Note that you can also click the **Project** tab and select any of these options for the `C1DockControl`.

If you click the pushpin , the `C1DockTabItems` will appear on the left and slide open when clicked.



If you click the **Hide** button , the `C1DockControl` will be hidden.

3. Select the **Toolbox** header and drag it over the `C1DockControl` with the `myfile.ext` tab. Notice the dock indicators that appear over the dock zones. The dock zones are shaded in blue when you hover over an indicator.



The second C1DockControl with the myfile.ext C1DockTabItem does not have a header since we set **ShowHeader** to **False**.

Congratulations! You have successfully completed the **DockControl for Silverlight** quick start. In this quick start, you've used the C1DockControl to create several windows that can float or be docked.

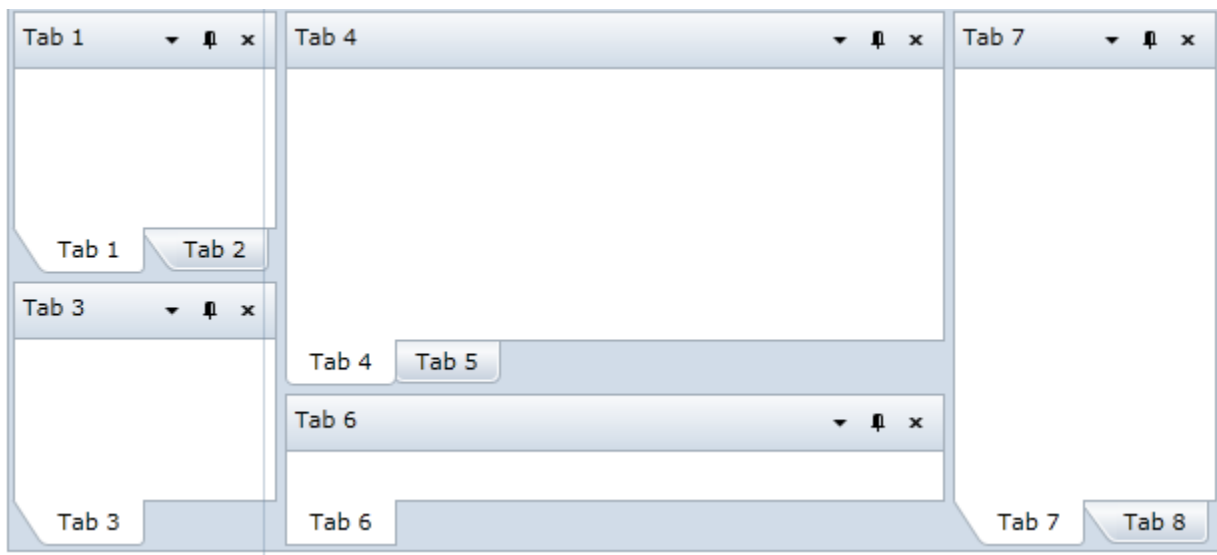
## XAML Quick Reference

This topic is dedicated to providing a quick overview of the XAML used to create a C1DockControl, C1DockGroup, C1DockTabControl, and C1DockTabItem.

To get started developing, add a reference to <http://schemas.componentone.com>:

```
xmlns:c1="http://schemas.componentone.com/winfx/2006/xaml"
```

Here is a sample docking layout:



Below is the XAML for the sample docking layout:

```
<c1:C1DockControl x:Name="dockControl" Margin="-128,0,12,127">
  <c1:C1DockGroup>
    <c1:C1DockTabControl Dock="Top">
      <c1:C1DockTabItem Header="Tab 1" TabShape="Sloped">
        <!--Your Content Here-->
      </c1:C1DockTabItem>
      <c1:C1DockTabItem Header="Tab 2" TabShape="Sloped">
        <!--Your Content Here-->
      </c1:C1DockTabItem>
    </c1:C1DockTabControl>
    <c1:C1DockTabControl>
      <c1:C1DockTabItem Header="Tab 3" TabShape="Sloped">
        <!--Your Content Here-->
      </c1:C1DockTabItem>
    </c1:C1DockTabControl>
  </c1:C1DockGroup>
  <c1:C1DockGroup>
```

```

        <c1:C1DockTabControl Dock="Top" DockWidth="500"
DockHeight="500" TabItemShape="Rounded">
            <c1:C1DockTabItem Header="Tab 4">
                <!--Your Content Here-->
            </c1:C1DockTabItem>
            <c1:C1DockTabItem Header="Tab 5">
                <!--Your Content Here-->
            </c1:C1DockTabItem>
        </c1:C1DockTabControl>
        <c1:C1DockTabControl>
            <c1:C1DockTabItem Header="Tab 6">
                <!--Your Content Here-->
            </c1:C1DockTabItem>
        </c1:C1DockTabControl>
    </c1:C1DockGroup>
    <c1:C1DockTabControl>
        <c1:C1DockTabItem Header="Tab 7" TabShape="Sloped">
            <!--Your Content Here-->
        </c1:C1DockTabItem>
        <c1:C1DockTabItem Header="Tab 8" TabShape="Sloped">
            <!--Your Content Here-->
        </c1:C1DockTabItem>
    </c1:C1DockTabControl>
</c1:C1DockControl>

```

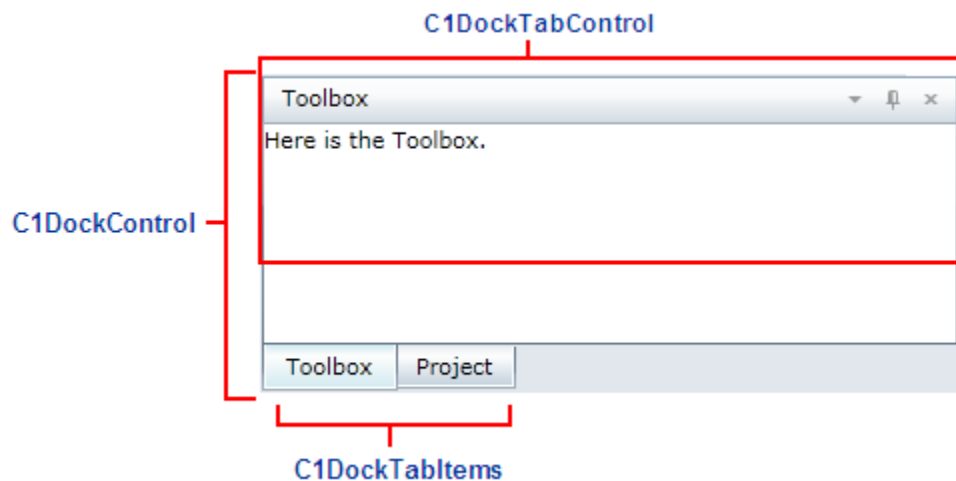
## Working with DockControl for Silverlight

**ComponentOne DockControl for Silverlight** includes the C1DockControl control, a simple control that allows you to dock, float, or tab windows. The following topics provide more information on the C1DockControl and the elements that can be added to it, options for docking windows, and docking indicators and zones.

### C1DockControl Elements

The C1DockControl serves as a container in which you can add other **DockControl for Silverlight** elements to help organize your windows.

Once you have a C1DockControl on the page, you can add a C1DockTabControl with C1DockTabItems.



The C1DockTabControl contains three buttons:

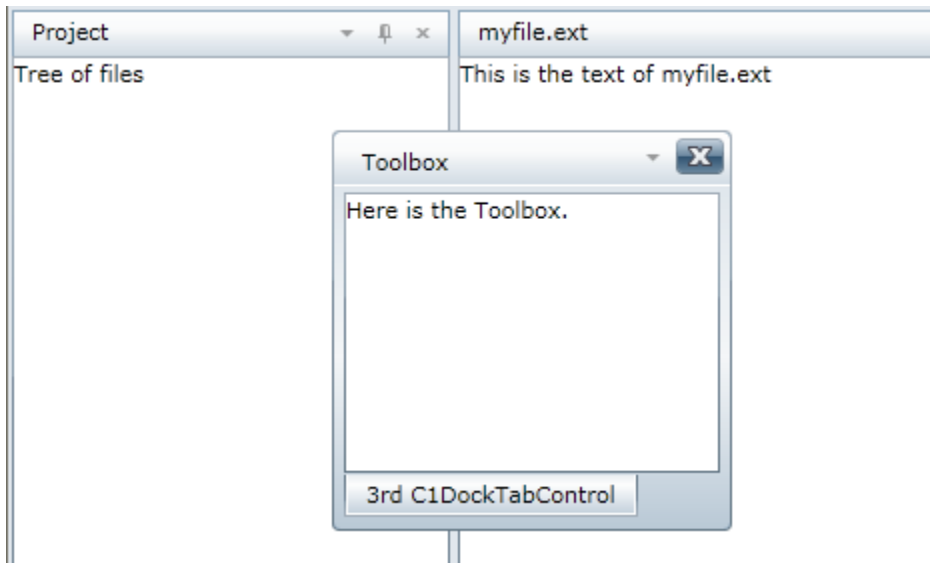
|                       |  |
|-----------------------|--|
| <b>Drop-down list</b> | DockMode options include: Floating, Docked, Sliding, Hidden  |
| <b>Pushpin</b>        | Enables the auto hide feature. The <b>C1DockTabItems</b> are minimized along the edges of the IDE. |
| <b>Hide</b>           | Hides the <b>C1DockTabControl</b> .  |

## Docking Options

**DockControl for Silverlight** provides four DockMode options: Floating, Docked, Sliding, and Hidden. The following images show each of the docking options available in the drop-down list of the C1DockTabControl.

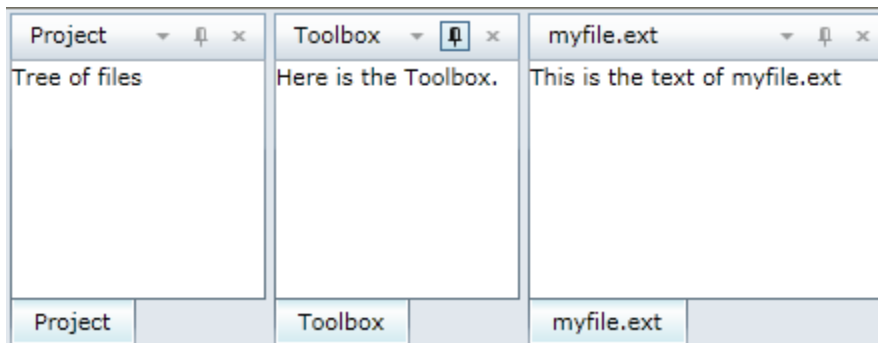
### Floating

Click **Floating** to undock the window and allow it to float over the other windows.



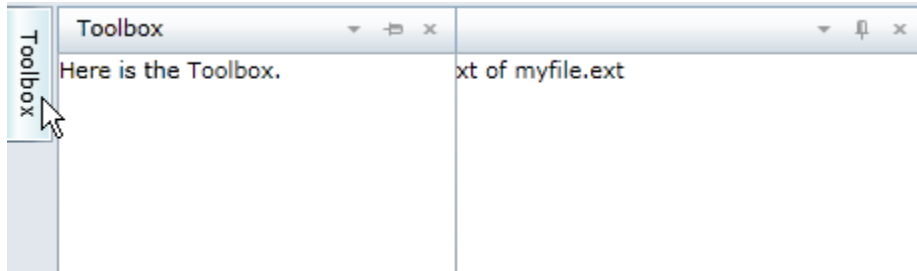
### Docked

Click **Docked** to dock the window with the other windows.



## Sliding

Click **Sliding** to minimize the `C1DockTabItem` along the edges of the IDE.



## Hidden

Click the **Hide** button to hide the `C1DockTabControl`.

### To set the Dock mode:

You can set the dock mode at design time using the `DockMode` property. The following XAML markup sets the dock mode to **Floating**:

```
<c1:C1DockTabControl DockMode="Floating">
  <c1:C1DockTabItem Header="Toolbox">
    <TextBlock Text="Toolbox" />
  </c1:C1DockTabItem>
</c1:C1DockTabControl>
```

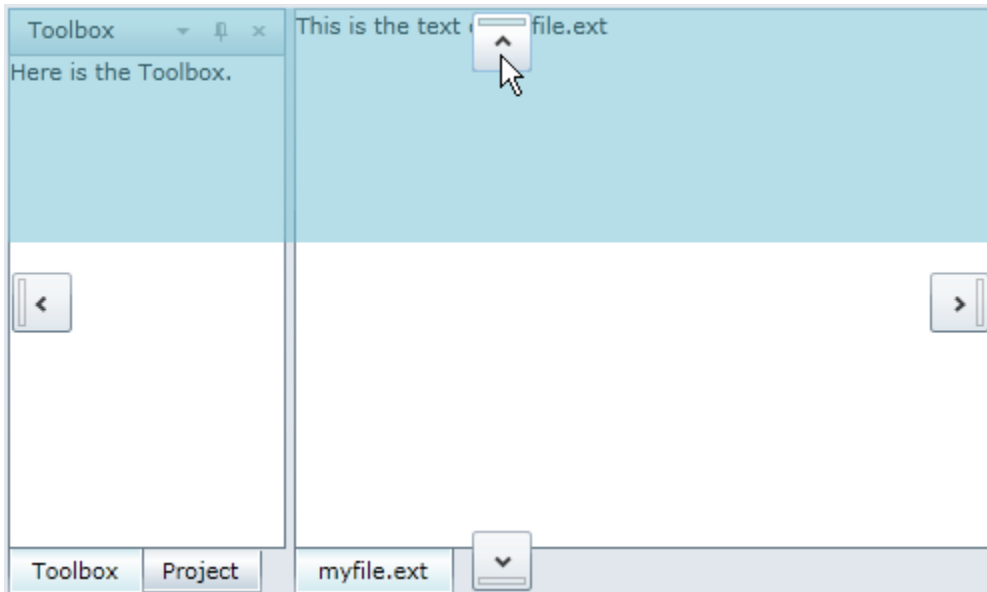
See [Setting the Dock Mode](#) (page 58) for more information.

## Docking Diamond and Zones

The docking diamond is used to show where `C1DockControl` elements can be docked.

The `C1DockControl` provides four docking indicators so you can dock at the top, right, bottom, or left side of the control.

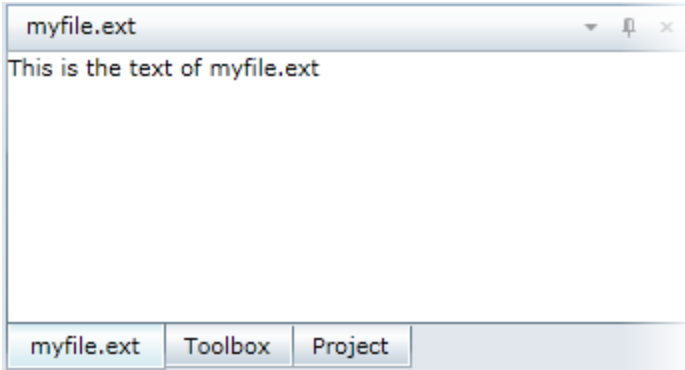
Drag the `C1DockTabControl` header, and blue docking zones will appear, showing you where you can dock the window.



If you have multiple **C1DockTabControl**s, drag one of the **C1DockTabControl** headers over one of the other **C1DockTabControl**s. You will notice the docking diamond shows five docking indicators, including top, right, bottom, left, and a center docking indicator that allows you to merge the **C1DockTabItems** into one **C1DockTabControl**.



Using the previous image's example, if the mouse is released over the center indicator, the *myfile.ext* **C1DockTabItem** will appear with the other **C1DockTabItems** in the first **C1DockTabControl**.



## DockControl for Silverlight Layout and Appearance

The following topics detail how to customize the **C1DockControl's** layout and appearance. You can use templates to format and layout the control and to customize the control's actions. Themes allow you to customize the appearance of the control and take advantage of Silverlight's XAML-based styling.

### Templates

One of the main advantages to using a Silverlight control is that controls are "lookless" with a fully customizable user interface. Just as you design your own user interface (UI), or look and feel, for Silverlight applications, you can provide your own UI for data managed by **ComponentOne DockControl for Silverlight**. Extensible Application Markup Language (XAML; pronounced "Zammel"), an XML-based declarative language, offers a simple approach to designing your UI without having to write code.

### Accessing Templates

You can access templates in Microsoft Expression Blend by selecting the C1DockControl and, in the **Object** menu, selecting **Edit Template**. Select **Edit a Copy** to create an editable copy of the current template or select **Create Empty** to create a new blank template.

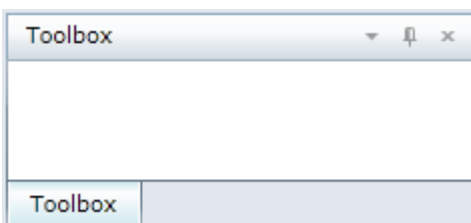
**Note:** If you create a new template through the menu, the template will automatically be linked to that template's property. If you manually create a template in XAML you will have to link the appropriate template property to the template you've created.

Note that you can use the [Template](#) property to customize the template.

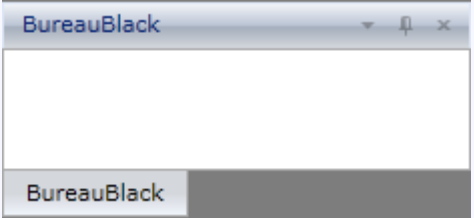

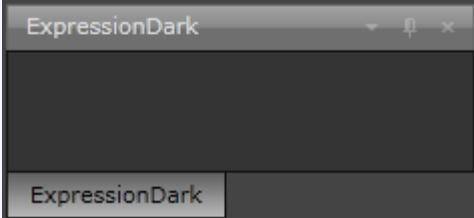
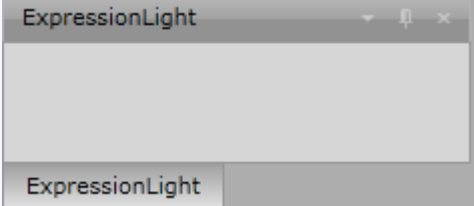
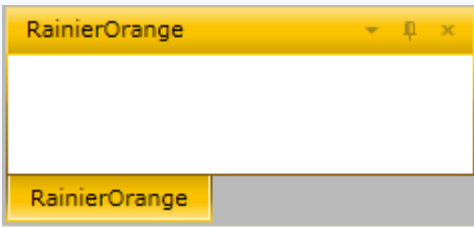
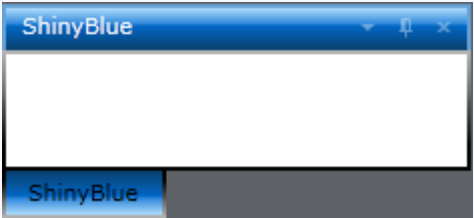
### DockControl Theming

Silverlight themes are a collection of image settings that define the look of a control or controls. The benefit of using themes is that you can apply the theme across several controls in the application, thus providing consistency without having to repeat styling tasks.

When C1DockControl is added to your project, it appears with the default theme, which looks similar to the following image:



But the C1DockControl can also be themed with one of our six included Silverlight themes: BureauBlack, ExpressionDark, ExpressionLight, RainierOrange, ShinyBlue, and WhistlerBlue. The table below provides a sampling of each theme.

| Full Theme Name        | Appearance   |
|------------------------|--|
| C1ThemeBureauBlack     |    |
| C1ThemeCosmopolitan    |    |
| C1ThemeExpressionDark  |   |
| C1ThemeExpressionLight |  |
| C1ThemeRainierOrange   |  |
| C1ThemeShinyBlue       |  |



You can add any of these themes to a **C1DockControl** by declaring it around the control in XAML markup. For more information about adding themes, see the [Using Silverlight Themes](#) (page 59).

## DockControl ClearStyle Properties

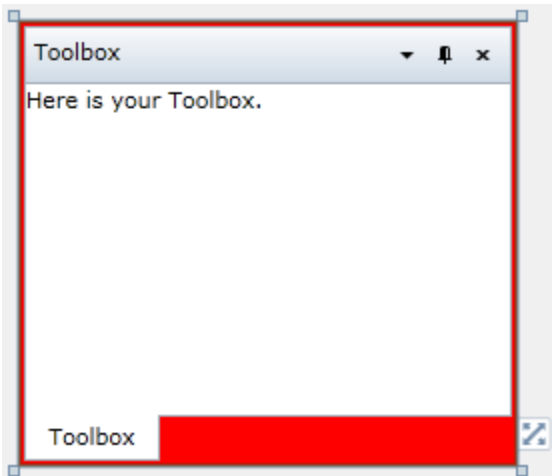
**DockControl for Silverlight** supports ComponentOne's ClearStyle technology, which allows you to easily change control colors without having to change control templates. By setting a few color properties, you can quickly style the **C1DockControl** elements. The supported properties for **C1DockControl** are listed in the following table:

| Property             | Description   |
|----------------------|---|
| Background           | Gets or sets the background used to fill the <b>C1DockControl</b> .                           |
| MouseOverBrush       | Gets or sets the brush used to highlight the control when the mouse is hovering over it.      |
| TabControlBackground | Gets or sets the background color of the <b>C1DockTabControl</b> .                            |
| TabControlForeground | Gets or sets the foreground color, or the color of the text, in the <b>C1DockTabControl</b> . |
| TabStripBackground   | Gets or sets the background color of the <b>C1DockTabItem</b> .                               |
| TabStripForeground   | Gets or sets the foreground color, or the color of the text, in the <b>C1DockTabItem</b> .    |

You can completely change the appearance of the **C1DockControl** by setting these properties. For example, if you set the **C1DockControl.Background** property to **Red** so the XAML markup appears similar to the following:

```
<c1:C1DockControl Background="Red" Name="c1DockControl1" >
  <c1:C1DockTabControl Dock="Left">
    <c1:C1DockTabItem Header="Toolbox">
      <TextBlock Text="Here is your Toolbox." />
    </c1:C1DockTabItem>
  </c1:C1DockTabControl>
</c1:C1DockControl>
```

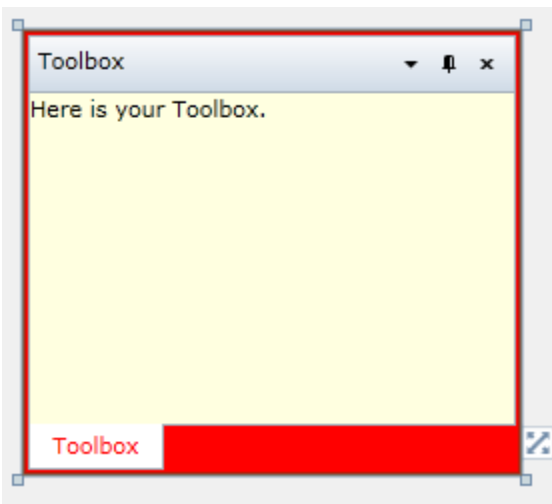
The **C1DockControl** will look similar to the following:



Experiment with the other properties to quickly change the look of the **C1DockControl** elements. For example, the following XAML sets the **Background**, **TabStripForeground**, and **TabControlBackground** properties:

```
<c1:C1DockControl Background="Red" TabStripForeground="Red"
TabControlBackground="LightYellow" Name="c1DockControl1" >
  <c1:C1DockTabControl Dock="Left">
    <c1:C1DockTabItem Header="Toolbox">
      <TextBlock Text="Here is your Toolbox." />
    </c1:C1DockTabItem>
  </c1:C1DockTabControl>
</c1:C1DockControl>
```

The **C1DockControl** will look like this:



## DockControl for Silverlight Task-Based Help

The task-based help assumes that you are familiar with programming in Visual Studio .NET and know how to use the C1DockControl in general. If you are unfamiliar with the **ComponentOne DockControl for Silverlight** product, please see the [DockControl for Silverlight Quick Start](#) (page 45) first.

Each topic in this section provides a solution for specific tasks using the **ComponentOne DockControl for Silverlight** product.

Each task-based help topic also assumes that you have created a new Silverlight project.

## Setting the Dock Mode

You can set the dock mode at design time using the DockMode property.

To set the dock mode, follow these steps:

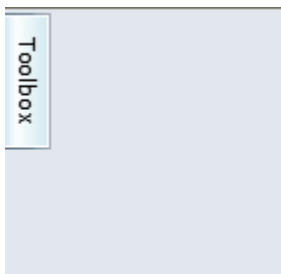
1. Open the .xaml page in Visual Studio.
2. Place your cursor between the <Grid></Grid> tags.
3. In the Toolbox, double-click the C1DockControl icon to add the control to the project.
4. Place your cursor between the <c1:C1DockControl> and </c1:C1DockControl> tags.
5. In the Toolbox, double-click the C1DockTabControl icon to add the control to the project.
6. Set the DockMode property to **Sliding**. Your XAML markup will now look similar to this:

```
<c1:C1DockControl>
  <c1:C1DockTabControl DockMode="Sliding"></c1:C1DockTabControl>
</c1:C1DockControl>
```

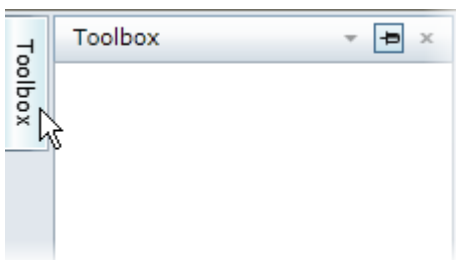
7. Place your cursor between the <c1:C1DockTabControl> and </c1:C1DockTabControl> tags.
8. In the Toolbox, double-click the C1DockTabItem icon to add the control to the project and set the **C1DockTabItem.Header** property to **Toolbox**. Your XAML should look similar to the following:

```
<c1:C1DockControl>
  <c1:C1DockTabControl DockMode="Sliding">
    <c1:C1DockTabItem Header="Toolbox"></c1:C1DockTabItem>
  </c1:C1DockTabControl>
</c1:C1DockControl>
```

9. Run your project. The C1DockControl will resemble the following image:



10. Click the **Toolbox** tab to slide the window into view.



## Using Silverlight Themes

You can apply six different themes to the C1DockControl. See [DockControl Theming](#) (page 54) for an example of each of the themes. In this topic, you will change the C1DockControl control's theme to C1ThemeRainierOrange.

### In Blend

Complete the following steps:

1. Click the **Assets** tab.
2. In the search bar, enter "C1ThemeRainierOrange". The C1ThemeRainierOrange icon appears.
3. Double-click the C1ThemeRainierOrange icon to add it to your project.
4. Under the **Objects and Timeline** tab, select [**C1ThemeRainierOrange**].
5. In the search bar, enter "C1DockControl ". The C1DockControl icon appears.
6. Drag the C1DockControl icon to the C1ThemeRainierOrange item in the **Objects and Timeline** tab.
7. Release the mouse button, and the C1DockControl is added as a child of C1ThemeRainierOrange.
8. In the search bar, enter "C1DockTabControl ". The C1DockTabControl icon appears.
9. Drag the C1DockTabControl icon to the C1DockControl item in the **Objects and Timeline** tab.
10. Release the mouse button, and the C1DockTabControl is added as a child of C1DockControl.
11. Press F5 to run your project and observe that the C1DockControl resembles the following:



### In Visual Studio

Complete the following steps:

1. Open the .xaml page in Visual Studio.
2. Place your cursor between the <Grid></Grid> tags.
3. In the Toolbox, double-click the **C1ThemeRainierOrange** icon to declare the theme. Its tags will appear as follows:

```
<my:C1ThemeRainierOrange></my:C1ThemeRainierOrange>
```

A reference to the C1.Silverlight.Theming and C1.Silverlight.Theming.RainierOrange assemblies will be added to your project.
4. Place your cursor between the <my:C1ThemeRainierOrange> and </my:C1ThemeRainierOrange> tags.
5. In the Toolbox, double-click the C1DockControl icon to add the control to the project. Its tags will appear as children of the <my:C1ThemeRainierOrange> tags, causing the markup to resemble the following:

```
<my:C1ThemeRainierOrange>  
  <c1:C1DockControl></c1:C1DockControl>  
</my:C1ThemeRainierOrange>
```
6. Place your cursor between the <c1:C1DockControl> and </c1:C1DockControl> tags.
7. In the Toolbox, double-click the C1DockTabControl icon to add the control to the project. Your XAML markup will now look similar to this:

```
<my:C1ThemeRainierOrange>  
  <c1:C1DockControl>  
    <c1:C1DockTabControl></c1:C1DockTabControl>  
  </c1:C1DockControl>  
</my:C1ThemeRainierOrange>
```

8. Press F5 to run your project and observe that the C1DockControl resembles the following:

