WPF及Silverlight版DockControl概述

WPF及Silverlight版DockControl可以帮助您在**WPF**应用程序中处理多个窗口，与**Microsoft Visual Studio**中的docking系统相似，**C1DockControl**提供给了停靠，浮动和标签窗口，您也可以设置自动隐藏区域以及轻松对**C1DockControl**风格化。

WPF及Silverlight版DockControl包括以下两个控件：

- C1DockControl
- C1DockTabControl

WPF及Silverlight版帮助文档

入门

- 关于 ComponentOne Studio WPF Edition安装，授权，技术支持，命名空间及使用控件创建工程，请参阅Getting Started with WPF Edition。
- 关于 ComponentOne Studio Silverlight Edition安装，授权，技术支持，命名空间及使用控件创建工程，请参阅Getting Started with Silverlight Edition。

C1DockControl主要特性

您可以从下述**C1DockControl**的主要特性中找到您需要的信息：

- 停靠指示器

  在默认情况下，**DockControl**采用的是类似于**Visual Studio**的Microsoft停靠指示器，更多信息请参阅**停靠指示器与区域**。

- 自然停靠

  自定义**DockControl**提供的自然停靠允许您在应用程序中停靠和浮动多个窗口，当您在停靠区域上拖拽窗口时，会出现可停靠的标识。

- 浮动窗口

  允许一个窗口浮动在一个独立的窗口应用上。

- 标签窗口

  在编辑器上打开文件，会在标签板上自动的排列。

- 自动隐藏

  如果在某一时刻需要看到更多代码，您可以使用图钉来自动隐藏窗口，这就允许您将不需要的工具窗口最小化到IDE的边框中。

- 丰富的API

  为了兼顾灵活性和制约性，每个停靠窗口的停靠状态都可以通过丰富的可编程API来控制。

- 支持两台监视(仅限WFP版)

  您可以轻松的从一台监视器拖拽选中的窗口到应一台监视器中，此项特性只限于**WPF**平台。

- 支持ClearStyle技术

  WPF及Silverlight版DockControl支持ComponentOne的ClearStyle技术，即允许您可以不用改变控件的模板就可以轻松的更改控件的颜色，只需要设置几个颜色的属性，您就可以快速的设置**C1DockControl**相关元素，想了解ComponentOne ClearStyle技术的相关信息，请参阅**DockControl ClearStyle属性**。.

C1DockControl快速入门

快速入门可以帮助您顺利的创建和运行**WPF及Silverlight版DockControl**。快速入门中，您需要在Visual Studio中创建一个新工程，添加一个C1DockControl到您的应用程序中，并在此基础上再添加一个附有C1DockTabItems的C1DockTabControl。

第一步：创建一个WPF应用程序

在这一步中，您将使用**WPF及Silverlight版DockControl**在Visual Studio中创建一个WPF或Silverlight应用程序。

您需要完成以下步骤来建立一个工程并添加一个**C1DockControl**控件到您的应用程序中：

1. 在Visual Studio中，选择**文件|新建|项目**，注意：完成快速入门的步骤需要Visual Studio2010及以上版本。
2. 在**新建项目**对话框中，在左侧控制板中选择语言（如C#），并在模板列表中选择**WPF或Silverlight**应用程序。
3. 在.NET Framework下拉列表中，选择**.NET Framework 3.5**及以上版本。
4. 输入**项目名称**并点击**确定**，此时会出现**新的WPF应用程序**或者**新的Silverlight应用程序**对话框。
5. 如果需要**为解决方案创建目录**，则可以选中复选框，点击确定**MainPage.xaml**文件将被打开。
6. 在XAML窗口中，将光标移至<Grid>与</Grid>标签之间并点击一次。
7. 导航到工具栏并拖拽**C1DockControl**图标到<Grid>标签之间，添加**C1DockControl**到**Window.xaml**中，产生的XAML标签将与下述相似：

WPF XAML

```
<Window 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:c1="http://schemas.componentone.com/winfx/2006/xaml"
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>
</Window>
```

Silverlight XAML

```
<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>
```

> **注意：**C1.WPF.Docking或C1.Silverlight.Docking的命名空间与*<c1:C1DockControl></c1:C1DockControl>*标签已经被添加到工程中。

在下一步中，您将添加一个附有**C1DockTabItems**的**C1DockTabControl**。

第二步：添加一个附有C1DockTabItems的C1DockTabControl

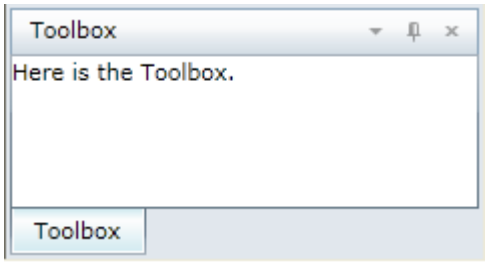
接下来，我们将添加一个附有**C1DockTabItems**的**C1DockTabControl**。

1. 在XAML标签中，将您的光标移至<c1:C1DockControl></c1:C1DockControl>之间并敲击回车键。
2. 在项目中，采用以下XAML标签添加一个**C1DockTabControl**：

XAML

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

此XAML中包含一个标题为**Toolbox**的**C1DockTabItem**，如果您运行应用程序，**C1DockTabControl**外观将与下图的显示相似：



让我们添加另一个**C1DockTabItem**和**C1DockTabControl**，此时当您运行应用程序时，可以选择**C1DockTabControl**的标签，停靠或者浮动窗口。

3. 请在XAML中</c1:C1DockTabItem>标签之后，添加另一个**C1DockTabItem**和**C1DockTabControl**，此时**C1DockControl**的XAML将与下述标签相似：

XAML

```
<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>
```

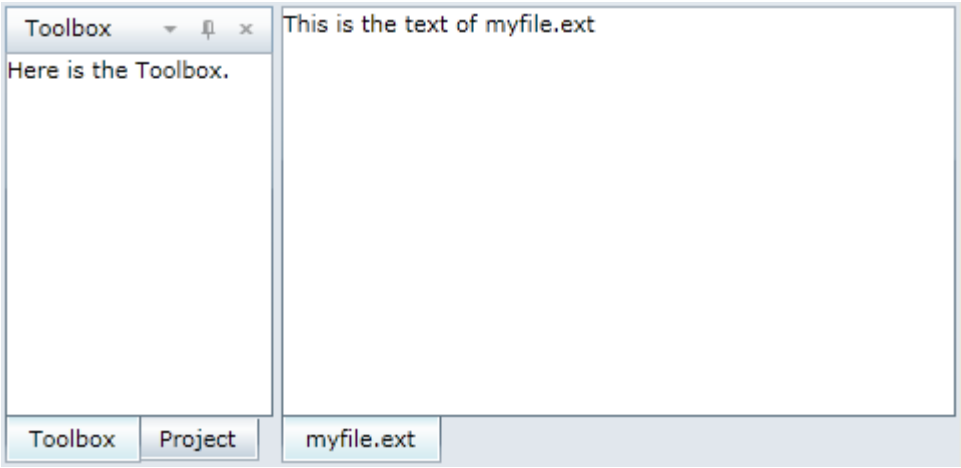
```
        <c1:C1DockTabControl DockWidth="500" ShowHeader="False">
            <c1:C1DockTabItem Header="myfile.ext">
                <TextBlock Text="This is the text of myfile.ext" />
            </c1:C1DockTabItem>
        </c1:C1DockTabControl>
    </c1:C1DockControl>
```

在下一步中您将运行应用程序及改变窗口的停靠模式。

第三步：运行应用程序

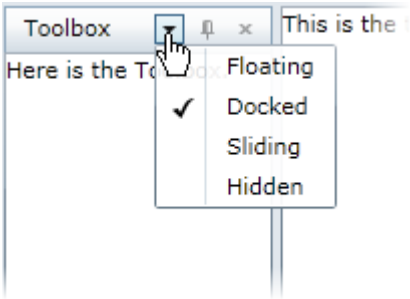
您已经成功创建了一个包含**C1DockControl**和多个标签项的**WPF或Silverlight**应用程序，接下来请您完成以下步骤运行应用程序：

1. 从菜单中选择**调试**，再选择**启动调试**，此时您可以预览应用程序的外观，其外观会与下图相似：



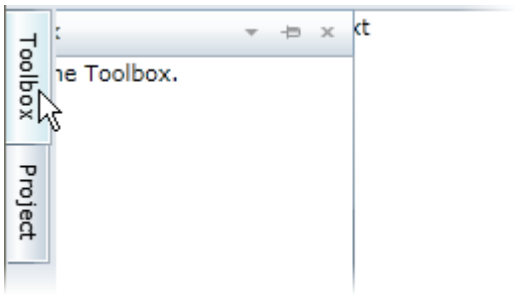
Toolbox停靠在左边，正如我们在XAML中的设定。

2. 点击**Toolbox**标题栏中的向下箭头选项，可以选择浮动、滑动或者隐藏**C1DockControl**选项。



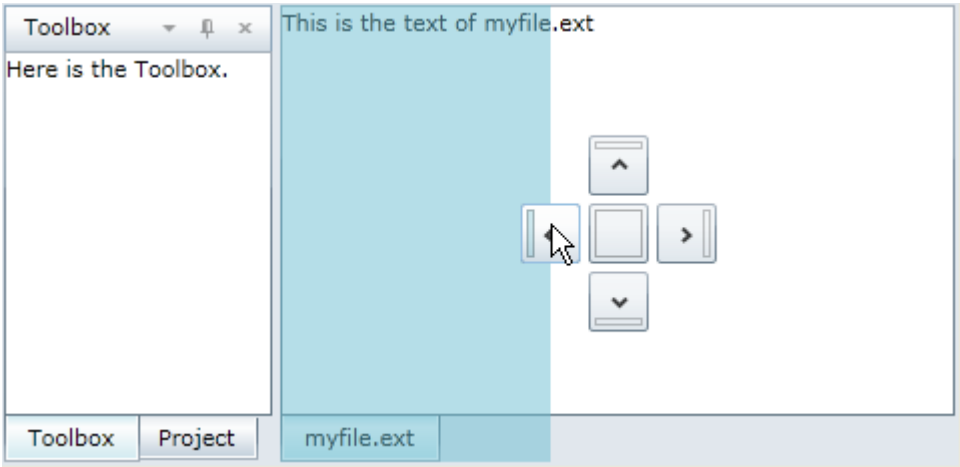
注意到您也可点击项目标签来选择任意的C1DockControl项。

如果您点击了**图钉按钮**，**C1DockTabItems**会出现在左侧。当再点击标签时它会被滑动打开。



如果您点击**隐藏**按钮 ， **C1DockControl**将会被隐藏。

3. 选择**Toolbox**标题栏，拖拽并覆盖标签为myfile.ext的**C1DockControl**，注意到停靠指示器会出现在停靠区域，当您来回拖动时停靠区域会出现蓝色的阴影效果。



其次需要注意的是，附有myfile.ext **C1DockTabItem**的**C1DockControl**没有标题，因此我们不能将设置**ShowHeader**为**False**。

恭喜您！您已经成功的完成了**WPF及Silverlight版DockControl**的快速入门，在快速入门中，您可以使用**C1DockControl**来创建几个可以浮动或者停靠的窗口。

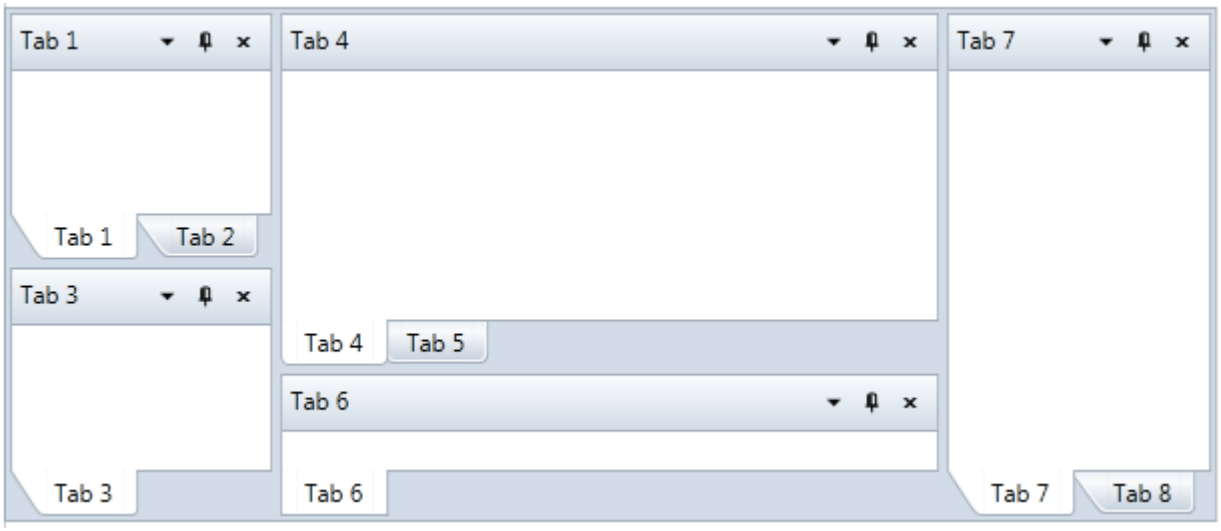
XAML快速参考

本主题主要给出了一个如何采用XAML创建**C1DockControl**、**C1DockGroup**、**C1DockTabControl**及**C1DockTabItem**的快速概要。

在开发之前，需要在根元素标签下添加一个C1命名声明：：

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

以下给出一个停靠的布局模板：



以下是停靠布局模板的XAML描述：

XAML

```
<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>
```

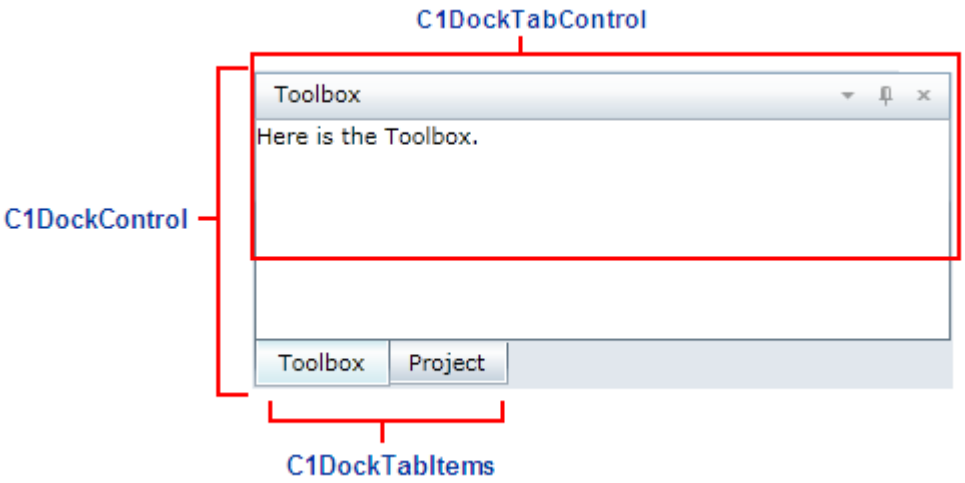
C1DockControl的使用

WPF及Silverlight版DockControl主要包含**C1DockControl**控件，它可以帮助您停靠、浮动或者标签化一个窗口。以下主题将提供C1DockControl的相关元素、停靠窗口选项、停靠指示及停靠区域的更多信息。

C1DockControl相关元素

C1DockControl作为一个容器可以帮助您添加**WPF及Silverlight版DockControl**的其他一些相关元素，通过这些元素可以帮助您更好的组织窗口。

当您在页面上添加了一个**C1DockControl**后，可以在此基础上再添加一个附有**C1DockTabItems**的**C1DockTabControl**。



C1DockTabControl包含了三个按钮：

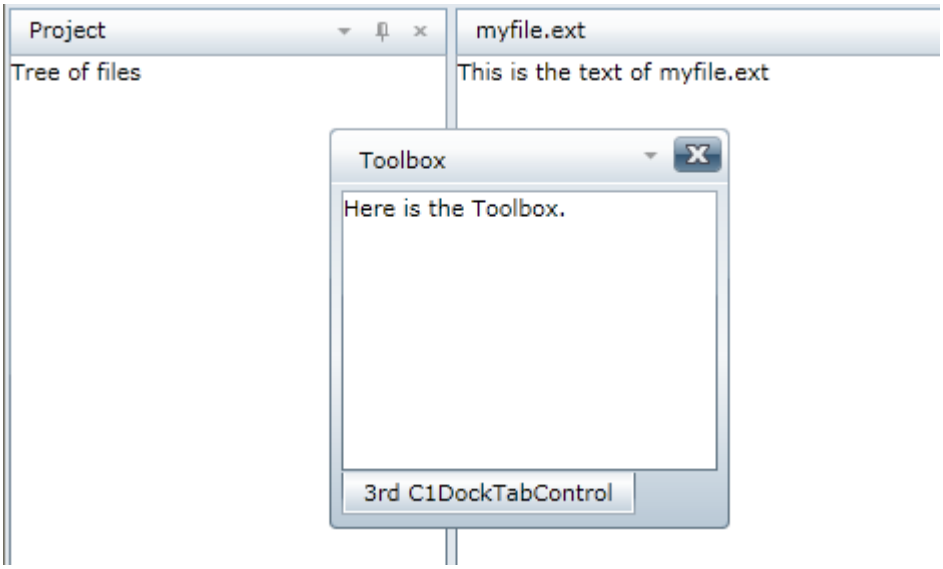
下拉菜单	停靠模式的选择包括：浮动，停靠，滑动，隐藏。
图钉	具有自动隐藏的特性，**C1DockTabItems**可以最小化到IDE的边框。
隐藏	隐藏**C1DockTabControl**。

停靠选项

WPF及Silverlight版DockControl提供四种停靠模式选项：浮动，停靠，滑动及隐藏。下图展示了**C1DockTabControl**下拉菜单中停靠选项的各种效果。

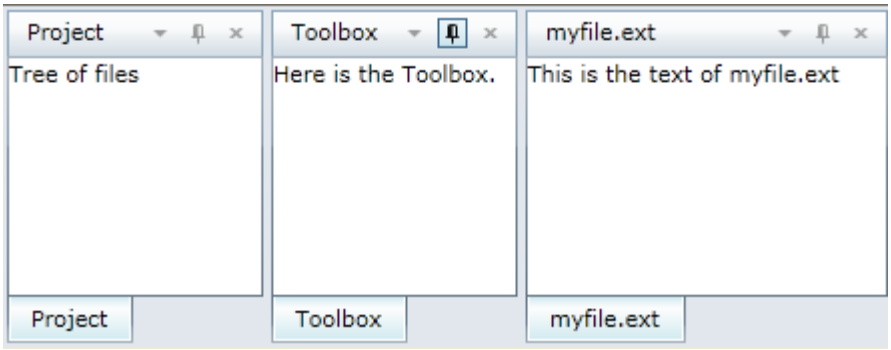
浮动

点击**浮动**按钮可以解除窗口的停靠，并允许它在其他的窗口上浮动。



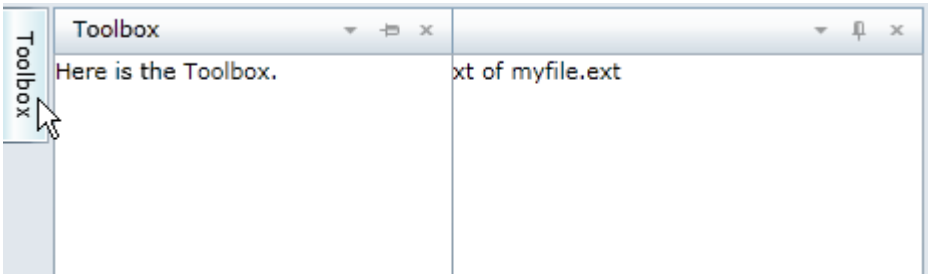
停靠

点击**停靠**按钮，可以将该窗口停靠在其他窗口之中。



滑动

点击**滑动**将**C1DockTabItem**最小化到IDE的边框。



隐藏

点击**隐藏**按钮来隐藏**C1DockTabControl**。

停靠模式的设置：

您在设计时可以通过**C1DockTabControl.DockMode**属性对停靠模式进行设置，以下是将停靠模式设置为**浮动**的XAML标签示例：

XAML

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

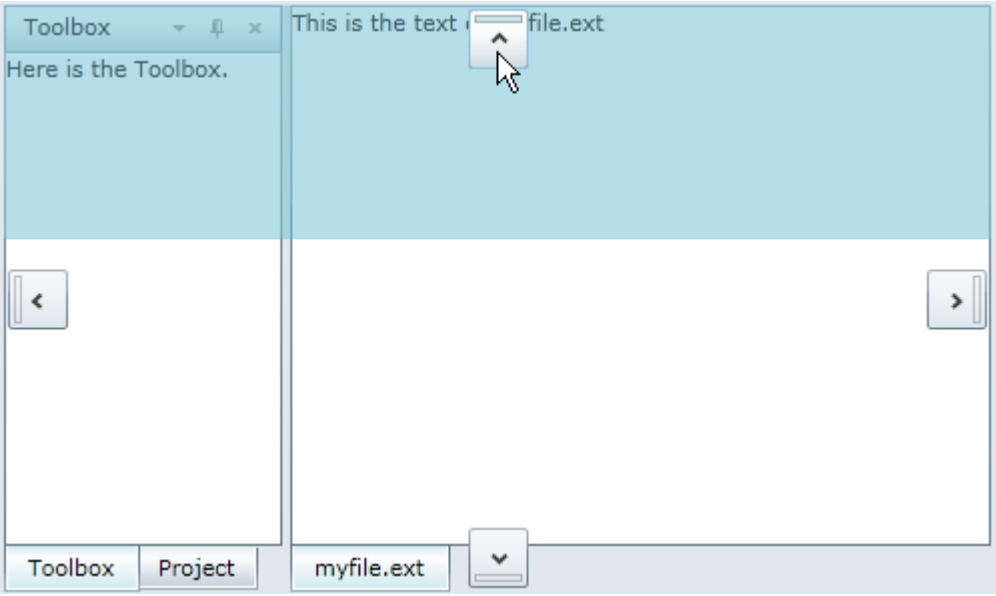
了解更多信息，请参阅**Dock模式的设置**。

停靠指示器与区域

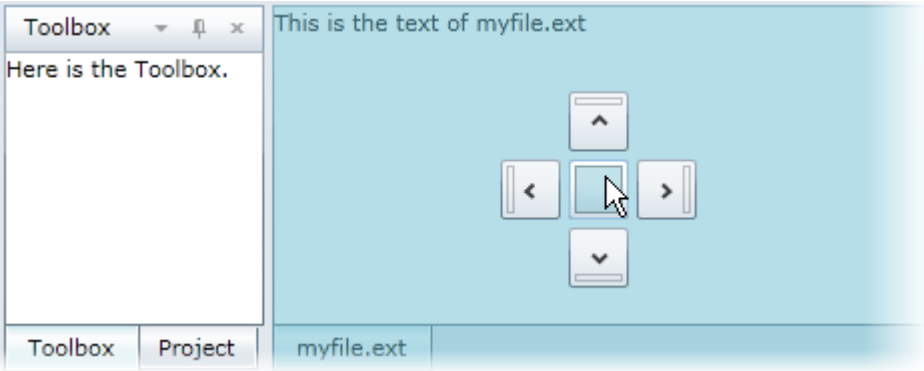
停靠指示器用来提示**C1DockControl**元素可以停靠的位置。

C1DockControl提供了四个停靠指示图标，因而您可以对上下左右的停靠方位进行控制。

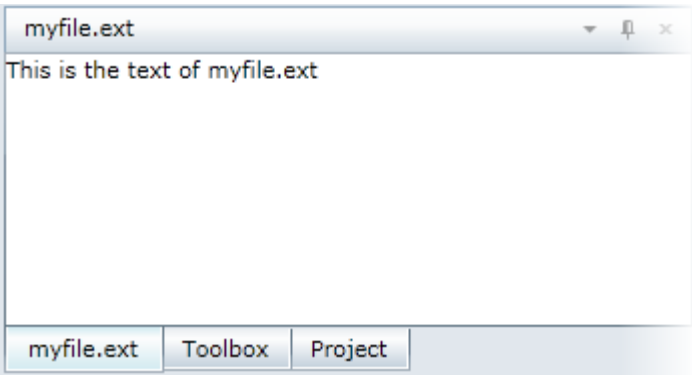
拖拽**C1DockTabControl**标题栏至您需要窗口停靠的位置，此时蓝色的停靠区域将会显示。



如果您需要多个**C1DockTabControl**,可以拖拽一个**C1DockTabControl**标题栏到其中一个您需要加入的**C1DockTabControl**上，此时您将会注意到停靠指示器会出现五个停靠指示图标，包括上方、右侧、下方、左侧及中央，这就允许您将多个**C1DockTabItem**合并到一个**C1DockTabControl**中。



参照上图的例子，如果鼠标在中央的指示图标上释放，myfile.ext **C1DockTabItem**将会同其他**C1DockTabItems**出现在第一个**C1DockTabControl**上。



C1DockControl布局和外观

以下主题将详细介绍如何自定义**C1DockControl**的布局和外观，您可以通过模板来定义控件的格式和布局，同时自定义控件的行为。

模板

WPF及Silverlight版控件的一个优势在于控件具有完全自定义的"无外观"用户界面，例如您在为WPF或Silverlight应用设计用户界面（UI）时，可以通过**WPF及Silverlight版DockControl**来实现适用于数据管理的UI。可扩展标记语言（XAML;发音为"Zammel"）是一种基于XML的声明式语言，它提供了一种简易的方式可以不用编写代码就可以完成您的UI设计。

可访问模板

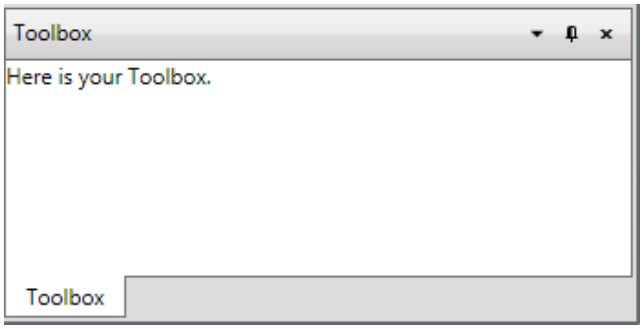
您可以访问Microsoft Expression Blend获取模板，选择**C1DockControl**并在**对象**菜单中选择**编辑模板**，通过选择**编辑拷贝**来创建当前模板可编辑的一个拷贝，或者选择创建一个新的空白模板。

> **注意：**如果您通过菜单方式创建了一个新的模板，模板将会自动的连接模板的属性，但如果您用XAML手动创建了一个模板，您就必须为您创建的模板提供适宜的模板属性。

注意您是通过模板属性来实现自定义模板。

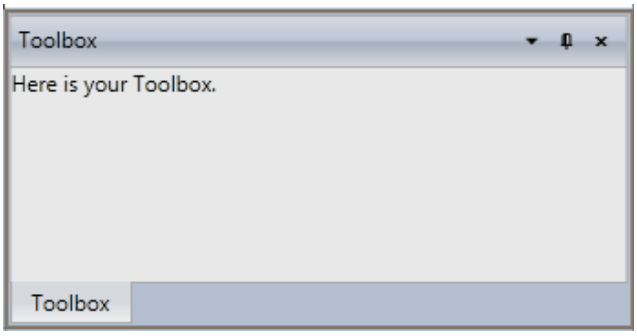
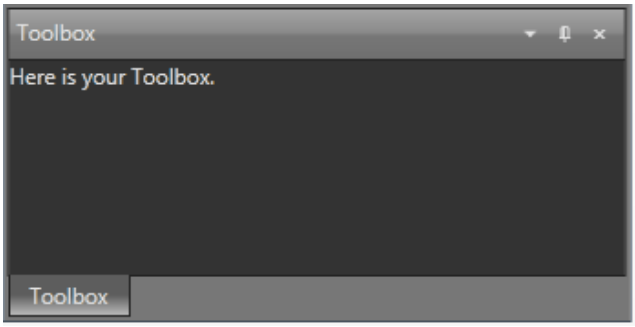
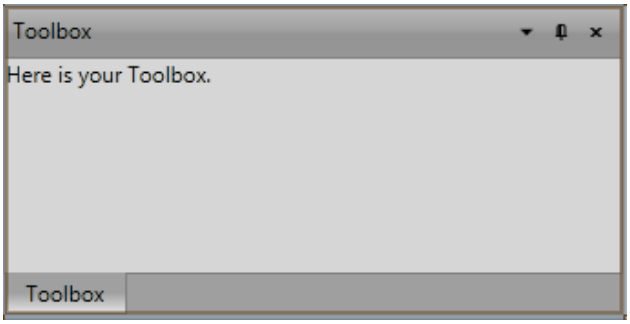
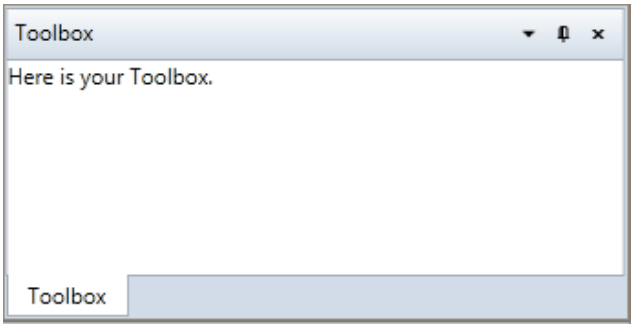
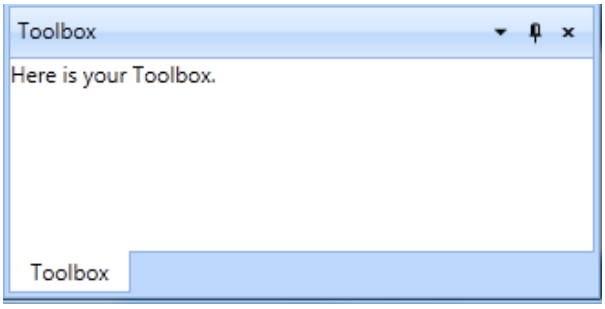
C1DockControl主题

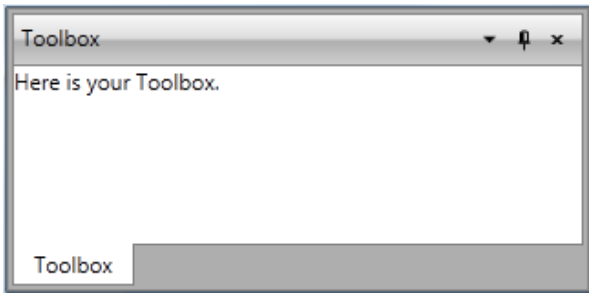
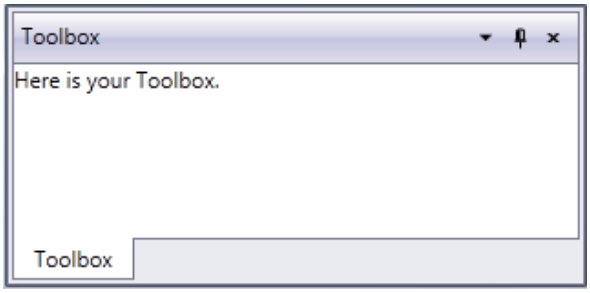
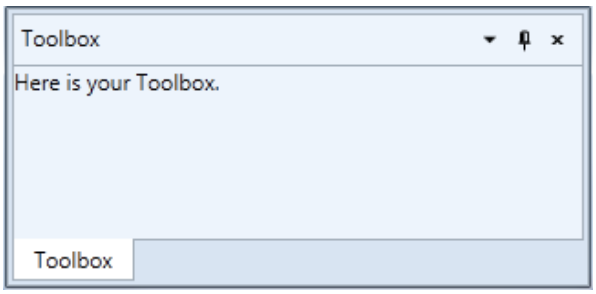
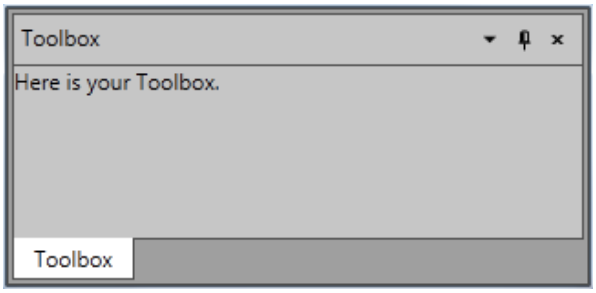
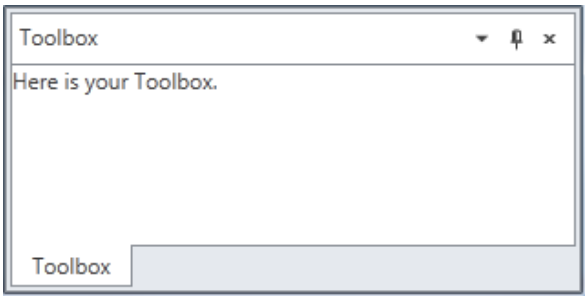
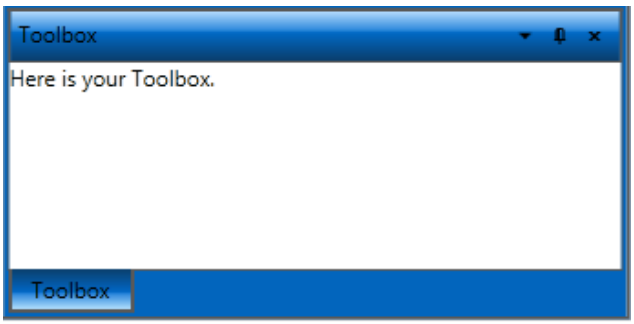
WPF版DockControl包含几种不同的主题允许您自定义网格外观，当您首次添加了一个包含**C1DockTabControl**和**C1DockTabItems**的**C1DockControl**控件时，它的呈现会与下图相类似：

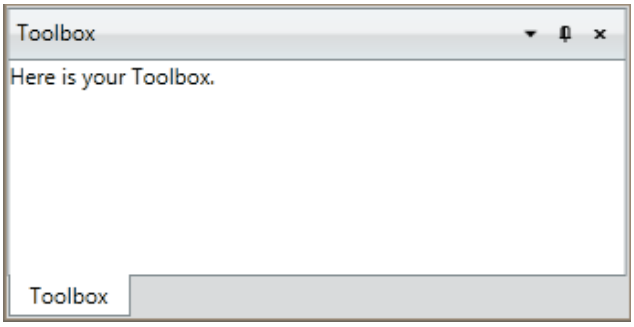


这是控件的默认外观，您可以通过使用内置主题或者已创建的自定义主题来改变控件的外观，所有的内置主题都是基于WPF Toolkit的主题，内置主题的描述和图示如下：

Theme Name	Theme Preview

C1ThemeBureauBlack	
C1ThemeExpressionDark	
C1ThemeExpressionLight	
C1Blue	
C1ThemeOffice2007Blue	

C1ThemeOffice2007Black	
C1ThemeOffice2007Silver	
C1ThemeOffice2010Blue	
C1ThemeOffice2010Black	
C1ThemeOffice2010Silver	
C1ThemeShinyBlue	

C1ThemeWhistlerBlue	

您需要通过使用**Apply Theme**方法来设置元素的主题，首先需要添加一个引用来完成您项目的主题配置，您可以采用下代码设置主题：

Visual Basic

```
Private Sub Window_Loaded(sender As System.Object, e As System.Windows.RoutedEventArgs)
Handles MyBase.Loaded
   Dim theme As New C1ThemeExpressionDark
   ' Using ApplyTheme
   C1Theme.ApplyTheme(LayoutRoot, theme)
```

C#

```
private void Window_Loaded(object sender, RoutedEventArgs e)
   {
      C1ThemeExpressionDark theme = new C1ThemeExpressionDark();
      //Using ApplyTheme
      C1Theme.ApplyTheme(LayoutRoot, theme);
   }
```

为了使主题应用于整个程序，您需要使用**System.Windows.ResourceDictionary.MergedDictionaries**属性，首先需要添加一个引用来完成您项目的主题配置，您可以用以下代码设置主题：

Visual Basic

```
Private Sub Window_Loaded(sender As System.Object, e As System.Windows.RoutedEventArgs)
Handles MyBase.Loaded
   Dim theme As New C1ThemeExpressionDark
   ' Using Merged Dictionaries
Application.Current.Resources.MergedDictionaries.Add(C1Theme.GetCurrentThemeResources(theme))
End Sub
```

C#

```
private void Window_Loaded(object sender, RoutedEventArgs e)
   {
      C1ThemeExpressionDark theme = new C1ThemeExpressionDark();
      //Using Merged Dictionaries
Application.Current.Resources.MergedDictionaries.Add(C1Theme.GetCurrentThemeResources(theme));
   }
```

注意此方法只有在第一次应用主题时有效，如果您想替换使用其他ComponentOne主题，需要将之前的主题从**Application.Current.Resources.MergedDictionaries**属性中移除。

ComponentOne ClearStyle技术

ClearStyle技术是一项新颖，快速且简捷的方法，它为**Silverlight及WPF版**控件提供风格支持，ClearStyle可以帮助您避免处理大量繁琐的XAML模板和样式资源，可以为控件创建一个自定义风格。

目前，如果您需要在所有标准的**WPF及Silverlight**控件上添加一个主题，就必须要创建一个风格资源模板，然而这个过程在**Microsoft Visual Studio**上完成是困难的，这就是为什么Microsoft要引入**Expression Blend**来简化此任务，但是对于那些不熟悉Blend平台或者没有时间学习的开发人员来说，两个平台的转换对于他们来说是一项挑战，此外您还需要一名设计人员，然而您的设计人员和开发人员需要共享XAML文件，这无疑也增加了开发的复杂性。

因此我们将ClearStyle技术引入，想以尽可能直观的方式引入到**Visual Studio**开发中。在大多数情况下，您可能只需要对应用中的控件做出简单的改变，而这个过程也需要尽可能简化。例如，你仅想改变数据网格的行的颜色，这里就只需要简单地设置一项属性，而不是创建一个完整而复杂的模板对颜色做出改变。

如何使用ClearStyle

控件风格的每个关键部分都应该是可以直接操作的，如一个简单的颜色属性，这就导致需要对每个控件制定一系列独特的风格属性。例如，一个**Gauge**有**PointerFill**和**PointerStroke**两种属性，而一个**DataGrid**具有**SelectedBrush**和**MouseOverBrush**两种行特性。

比方说，在您的窗体里有一个控件不支持ClearStyle，您可以通过ClearStyle创建XAML资源并以此匹配您窗体中的其他其他控件（如抓取确切的颜色），或者您想要利用ClearStyle覆盖风格的部分内容（如自定义的滚动条），这都可以通过ClearStyle实现，因为ClearStyle是可扩展并可以覆盖您需要的风格区域。

ClearStyle旨在成为一个快速、便捷的风格改变解决方案，但您依然可以采取老式的**ComponentOne**控件风格获取需方法，而且ClearStyle不会干预那些不常用的属性，那些属性通常需要完整的自定义设计。

DockControl ClearStyle属性

WPF及Silverlight版DockControl支持ComponentOne's ClearStyle技术，可以帮助您不用改变控件的模板就可以轻松的对控件的颜色做出更改。通过设置几个颜色的属性，您就可以快速的设置**C1DockControl**的风格，**C1DockControl**支持的属性如下表：

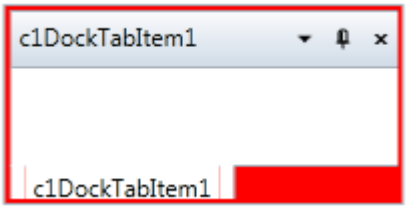
Property	Description
Background	获取或设置用来填充**C1DockControl**的背景颜色。
MouseOverBrush	获取或设置当鼠标略过控件时高亮的颜色。
TabControlBackground	获取或设置**C1DockTabControl**的背景色。
TabControlForeground	获取或设置**C1DockTabControl**的前景色或者文本的颜色。
TabStripBackground	获取或设置**C1DockTabItem**的背景色。
TabStripForeground	获取或设置**C1DockTabItem**的前景色或者文本颜色。

您可以通过设置这些属性完全的改变**C1DockControl**的外观，例如，如果您设置了**C1DockControl.Background**属性为**Red**，则XAML标签会与下述相似：

XAML

```
<my:C1DockControl Height="100" Background="Red" HorizontalAlignment="Left"
Margin="176,100,0,0" Name="c1DockControl1" VerticalAlignment="Top" Width="200">
            <my:C1DockTabControl Height="100" HorizontalAlignment="Left"
Name="c1DockTabControl1" VerticalAlignment="Top" Width="200">
                <my:C1DockTabItem Header="c1DockTabItem1" HorizontalAlignment="Left"
Name="c1DockTabItem1" VerticalAlignment="Top" />
            </my:C1DockTabControl>
        </my:C1DockControl>
```

C1DockControl的外观将与下图相似：

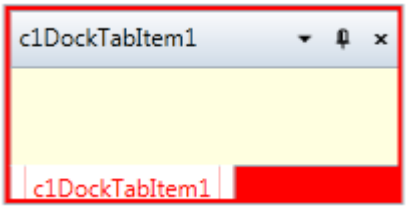


您可以通过其他属性来验证**C1DockControl**外观的快速改变，例如下述XAML设置了**Background**,**TabStripForeground**及**TabControlBackground**属性。

XAML

```
<my:C1DockControl Height="100" Background="Red" TabStripForeground="Red"
TabControlBackground="LightYellow" HorizontalAlignment="Left" Margin="176,100,0,0"
Name="c1DockControl1" VerticalAlignment="Top" Width="200">
            <my:C1DockTabControl Height="100" HorizontalAlignment="Left"
Name="c1DockTabControl1" VerticalAlignment="Top" Width="200">
                <my:C1DockTabItem Header="c1DockTabItem1" HorizontalAlignment="Left"
Name="c1DockTabItem1" VerticalAlignment="Top" />
            </my:C1DockTabControl>
        </my:C1DockControl>
```

C1DockControl的外观会与下图相似：



C1.WPF.Docking示例

请注意，本ComponentOne软件工具附有不同的示例工程或者demo，您可以利用包含在ComponentOne Studio中的开发工具来完成。

您可以参考下列路径中的产品示例：

Documents\ComponentOne Samples\WPF

ControlExplorer示例中的页面安装在**WPF版C1DockControl**详细功能中：

C# Sample

Sample	Description
DockingSamples	描述一个停靠窗口的基本示例

C1.Silverlight.DockControl示例

以下示例安装在默认示例目录的C1.Silverlight.Docking文件夹下。

Sample	Description
Blend Sample	**Blend**示例默认安装在示例目录下的**C1.Silverlight.Docking\Blend**文件夹下。 此示例利用以C1.Silverlight.Docking.dll配置的停靠及标签控件，创建了一个类似于Microsoft Expression Blend的环境。
C1Docking_Demo Sample	**C1Docking_Demo**示例默认安装在示例目录下的**C1.Silverlight.Docking\C1Docking_Demo**文件夹下。 此示例利用以C1.Silverlight.Docking.dll配置的停靠及标签控件，实现在停靠窗口填满整个应用窗口的场景下，滑出显示内容以及在浮动窗口内显示内容的效果。
C1Docking_Localization	**C1Docking_Localization**示例默认安装在示例目录下的**C1.Silverlight.Docking\C1Docking_Localization**文件夹下。 此示例演示了如何内置以C1.Silverlight.Docking.dll配置的控件。
VSDev9 Sample	**VSDev9**示例默认安装在示例目录下的**C1.Silverlight.Docking\VSDev9**文件夹下。 此示例利用以C1.Silverlight.Docking.dll配置的停靠及标签控件，创建了一个类似于Visual Studio的环境。

WPF版DockControl任务帮助

如果您熟悉Visual Studio编程且了解在一般情况下如何使用**C1DockControl**，请直接参考任务帮助，如果您是初次使用**WPF及Silverlight版DockControl**产品，请先参阅**C1DockControl快速入门**。

在使用**WPF及Silverlight版DockControl**产品时，每个任务帮助主题都提供一个特定任务的解决方案。

在您参阅任务帮助主题前，请确定您已经创建了一个新的**WPF或Silverlight**项目。

Dock模式设置

您可以在设计时使用**C1DockTabControl.DockMode**属性来设置Dock模式。

通过以下步骤设置Dock模式:

1. 在Visual Studio中打开.xaml页面。
2. 将光标置于*<Grid>*与*</Grid>*标签之间。
3. 双击Toolbox中的**C1DockControl**图标，将控件添加到项目中。
4. 将光标置于*<c1:C1DockControl>*与*</c1:C1DockControl>*之间。
5. 双击Toolbox中的**C1DockTabControl**图标，将控件添加到项目中。
6. 将**DockMode**属性设置为**Sliding**，此时您的XAML标签将会与下述相似：

XAML

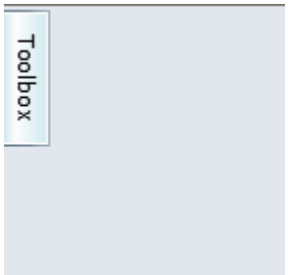
```
<c1:C1DockControl>
        <c1:C1DockTabControl DockMode="Sliding"></c1:C1DockTabControl>
</c1:C1DockControl>
```

7. 请将光标置于*<c1:C1DockTabControl>*与*</c1:C1DockTabControl>*标签之间。
8. 双击Toolbox中的**C1DockTabItem**图标，将控件添加到项目中，并设置**C1DockTabItem.Header**属性为**Toolbox**，此时您的XAML将会与下述相似：：

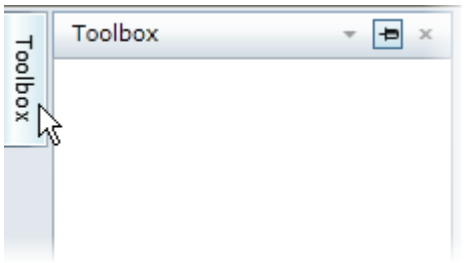
XAML

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

9. 运行您的项目，**C1DockControl**将与下图相似：



10. 点击**Toolbox**标签可以将窗口滑动到视图中。



如何禁用某些Dock位置

您可以禁用在**PickerLoading**事件中用户可以停靠的某些方向，例如通过以下步骤您可以禁用**C1DockControl**的左外边框停靠：

1. 首先添加一个**C1DockControl**和几个子**C1DockTabControl**到您的页面中。
2. 在**C1DockControl**上绑定**PickerLoading**事件。

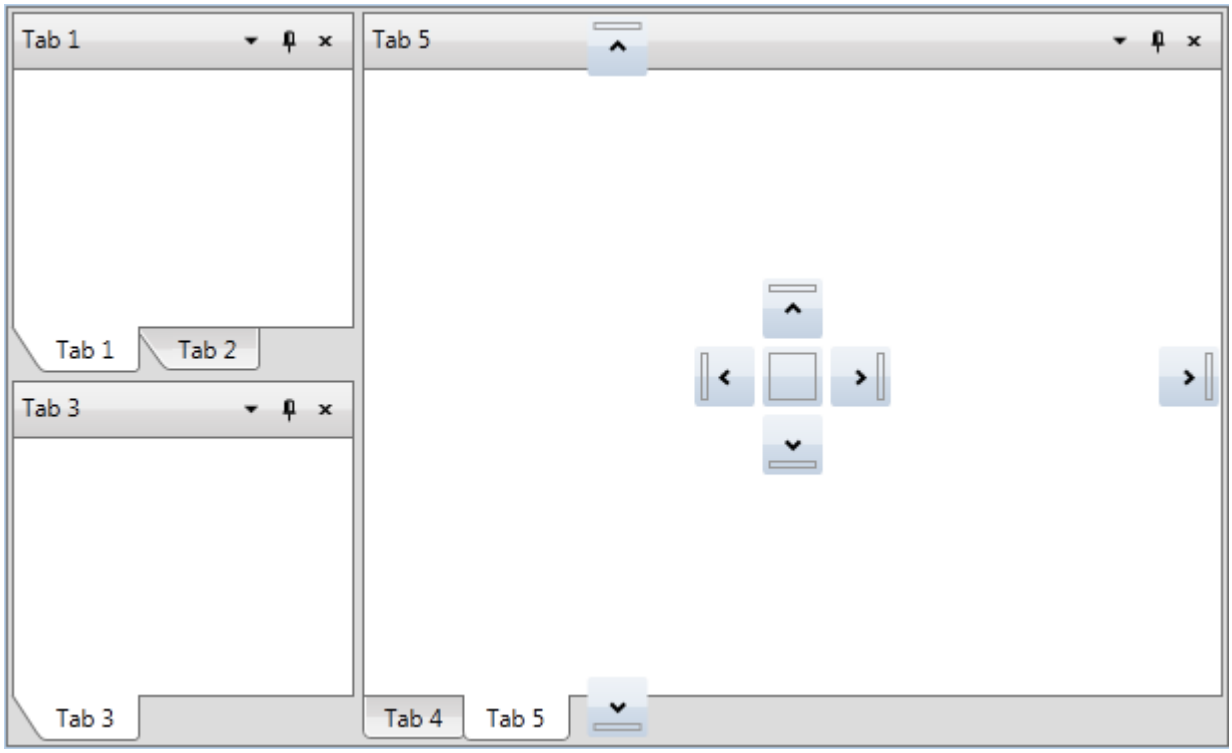
XAML

```
<c1:C1DockControlName="dockControl1"PickerLoading="dockControl1_PickerLoading"
/>
```

3. 在代码中，通过**PickerLoadingEventArgs**属性设置您想要禁用的停靠选择器数量，例如只要将**e.ShowLeftOuterPart**属性设置为**False**，您就可以禁用左外边框停靠。

C#

```
private void dockControl1_PickerLoading(object sender, PickerLoadingEventArgs e)
{
    // disable docking on the outer left edge
    e.ShowLeftOuterPart = false;
}
```


Tab 1
Tab 1
Tab 2
Tab 3
Tab 3
Tab 5
Tab 4
Tab 5