TileView for WPF/Silverlight

2018.02.20 更新

グレープシティ株式会社

目次

製品の概要

TileView for WPF/Silverlight を使用して、データを対話式に参照します。タイルを展開または折りたたんで、情報を表示する量を増減します。この高度なビジュアル化機能 と 対話式機能を備えたコントロールを使用して、アプリケーションで WPF/Silverlight の実力を示すことができます。ダッシュボード、詳細ビュー、フォトギャラリーなどを作成してください。

> **メモ**:説明内に含まれるクラスおよびメンバーに対するリファレンスへのリンクは、原則としてWPF版のリファレンスページを参照します。Silverlight版については、目次から同名のメンバーを参照してください。

主な特長

TileView for WPF/Silverlight を使用すると、機能豊富でカスタマイズされたアプリケーションを作成できます。以下に示す主要な機能をうまく活用して、TileView for WPF/Silverlight を最大限に使用してください。

- **タイルの展開と折りたたみ**

  C1TileView では、各タイルの状態を完全に制御できます。タイルを展開（最大化）または折りたたむ（最小化）ことができます。各タイルの状態によって情報の表示量が増減します。

- **最小化位置**

  1つのプロパティを設定するだけで、項目を C1TileView の上端、左端、下端、または右端に最小化できます。デフォルト状態での行数と列数を指定することもできます。

- **データ連結**

  C1TileView は、ビジネスオブジェクトの任意のコレクションに連結できる項目コントロールです。さまざまなデータテンプレートを設計して、各状態で表示できるデータ量を決定できます。

- **仮想化**

  C1TileView は、UI の仮想化をサポートしているため、大量の項目を即座にロードできます。

- **ドラッグアンドドロップインターフェイス**

  タイルがデフォルト状態の場合は、タイルをドラッグアンドドロップして並べ替えることができます。

- **アニメーション**

  C1TileView には、タイルが展開または折りたたまれたときのアニメーション効果が組み込まれています。

クイックスタート

このクイックスタートは、**TileView for WPF/Silverlight** を初めて使用するユーザーのために用意されています。このクイックスタートでは、**C1TileView** コントロールを使用して、簡単なプロジェクトを作成します。新しい WPF/Silverlight アプリケーションを作成し、**C1TileView** コントロールをアプリケーションに追加します。**C1TileView** コントロールに表示されるコンテンツを追加し、**TileView for WPF/Silverlight** に対して実行可能ないくつかの操作を確認してみます。

手順 1:アプリケーションの作成

この手順では、**TileView for WPF/Silverlight** を使用して WPF/Silverlight アプリケーションを作成します。アプリケーションに **C1TileView** コントロールを追加すると、コンテンツを内部に表示できるインターフェイスになります。プロジェクトをセットアップし、**C1TileView** コントロールをアプリケーションに追加するには、次の手順に従います。

1. Visual Studio で新しい WPF/Silverlight プロジェクトを作成します。この例では、アプリケーションに "QuickStart" という名前を付けます。プロジェクトに別の名前を付けた場合は、後の手順で "QuickStart" を参照している箇所を実際のプロジェクトの名前に変更する必要があります。
2. ソリューションエクスプローラで、プロジェクト名を右クリックし、**[参照の追加]**を選択します。**[参照の追加]**ダイアログボックスで、アセンブリ を見つけて選択し、**[OK]**をクリックして、プロジェクトに参照を追加します。
   - **C1.WPF** および **C1.Silverlight**
   - **C1.WPF.TileView** および **C1.Silverlight.TileView**
3. MainWindow.xaml ファイルの XAML ビューを開きます。このクイックスタートでは、XAML マークアップを使用して **C1TileView** コントロールを追加します。
4. 次のマークアップを使用して、XAML 名前空間を Window タグに追加します。

XAML

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

WPF

XAML

```
<Window x:Class="MainWindow"
xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
xmlns:c1="http://schemas.componentone.com/winfx/2006/xaml"
Title="TileView for WPF" Height="275" Width="425">
```

Silverlight

XAML

```
<UserControl x:Class="C1SilverlightCS111010.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"
xmlns:c1="http://schemas.componentone.com/winfx/2006/xaml"
```

```
mc:Ignorable="d" d:DesignHeight="262" d:DesignWidth="399">
```

これは、複数の名前空間を追加しなくても、ほとんどの ComponentOne WPF コントロールを使用できるようにするための統合名前空間です。

5. ページの Grid タグ内に <c1:C1TileView x:Name="C1TileView1"" /> タグを追加して、アプリケーションに **C1TileView** コントロールを追加します。XAML は次のようになります。

XAML

```
<Grid>
    <c1:C1TileView x:Name="C1TileView1"" />
</Grid>
```

これで、"C1TileView1" という名前の C1TileView コントロールがアプリケーションに追加されます。

これで、アプリケーションのユーザーインターフェイスが正しくセットアップされましたが、このアプリケーションを実行すると、C1TileView コントロールにコンテンツがないことがわかります。次の手順では、C1TileView コントロールにコンテンツを追加し、コントロールに対して実行可能ないくつかの操作を確認してみます。

手順 2：コントロールのカスタマイズ

前の手順では、WPF/Silverlight アプリケーションを作成し、プロジェクトに**C1TileView** コントロールを追加しました。アプリケーションをカスタマイズするには、次の手順に従います。

1. ページの **C1TileView** タグ内に AllowDrop="True" を追加して、ユーザーがコントロール内の項目に対してドラッグアンドドロップ操作を実行できるようにします。XAML マークアップは次のようになります。

XAML

```
<c1:C1TileView x:Name="C1TileView1"
            AllowDrop="True"></c1:C1TileView>
```

2. **C1TileView** タグ内に3つの**C1TileViewItem** を追加します。XAML マークアップは次のようになります。

XAML

```
<c1:C1TileView Name="C1TileView1" AllowDrop="True">
    <c1:C1TileViewItem></c1:C1TileViewItem>
    <c1:C1TileViewItem></c1:C1TileViewItem>
    <c1:C1TileViewItem></c1:C1TileViewItem>
</c1:C1TileView>
```

3. 各 **C1TreeViewItem** に **Background** プロパティと **Header** プロパティを追加します。マークアップは次のようになります。

XAML

```
<c1:C1TileView Name="C1TileView1" AllowDrop="True">
    <c1:C1TileViewItem Background="Red" Header="赤色"></c1:C1TileViewItem>
    <c1:C1TileViewItem Background="Blue" Header="青色"></c1:C1TileViewItem>
    <c1:C1TileViewItem Background="Yellow" Header="黄色"></c1:C1TileViewItem>
</c1:C1TileView>
```

これで、各項目が異なる色で表示され、ヘッダーにテキストが表示されます。

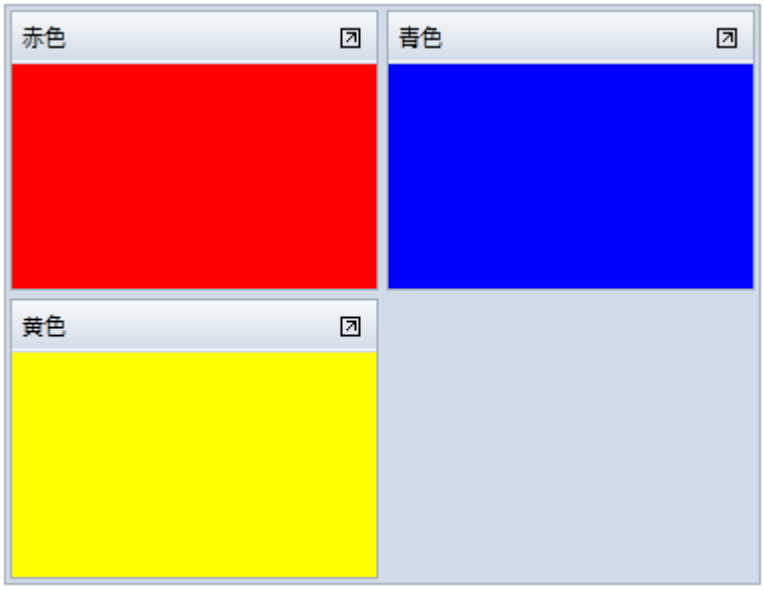
この手順では、C1TileView コントロールにコンテンツを追加しました。次の手順では、このコントロールで可能な実行時の操作

をいくつか示します。

手順 3:アプリケーションの実行

WPF アプリケーションを作成し、**C1TileView** コントロールをカスタマイズしました。次に、アプリケーションを実行します。アプリケーションを実行して **TileView for WPF/Silverlight** の実行時の動作を確認するには、次の手順に従います。

1. **[デバッグ]**メニューから**[デバッグ開始]**を選択し、実行時にアプリケーションがどのように表示されるかを確認します。



C1TileView コントロール内に3つの C1TileViewItem が表示されます。

2. 赤色の項目のヘッダーをクリックし、青色の項目に向かってドラッグします。項目が入れ替わります。
3. 黄色の項目のヘッダーの右上隅にある最大化アイコンをクリックします。他の2つの項目は最小化されます。

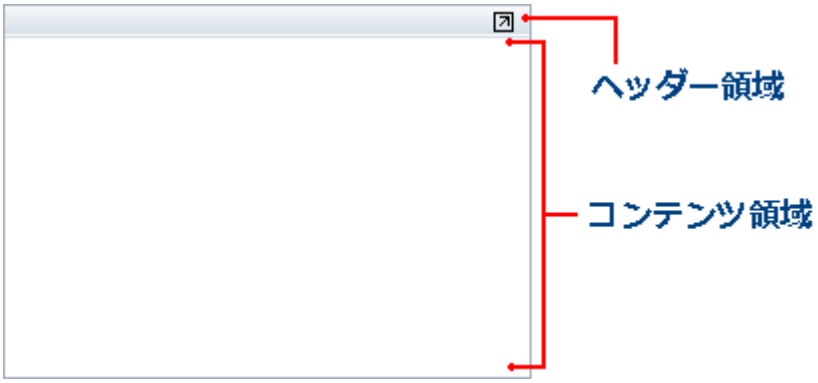
おめでとうございます。これで **TileView for WPF/Silverlight** クイックスタートは完了です。簡単な WPF アプリケーションを作成し、**TileView** コントロールを1つ追加してカスタマイズしました。その後、コントロールの実行時機能をいくつか確認しました。<

TileView の使い方

TileView for WPF/Silverlight には**C1TileView** コントロールが含まれます。これは、データを対話式に参照するためのパネルです。XAML ウィンドウに追加された**C1TileView** コントロールは、空のコンテナコントロールになります。このコントロールをカスタマイズしたり、コンテンツをロードすることができます。

TileViewItem の要素

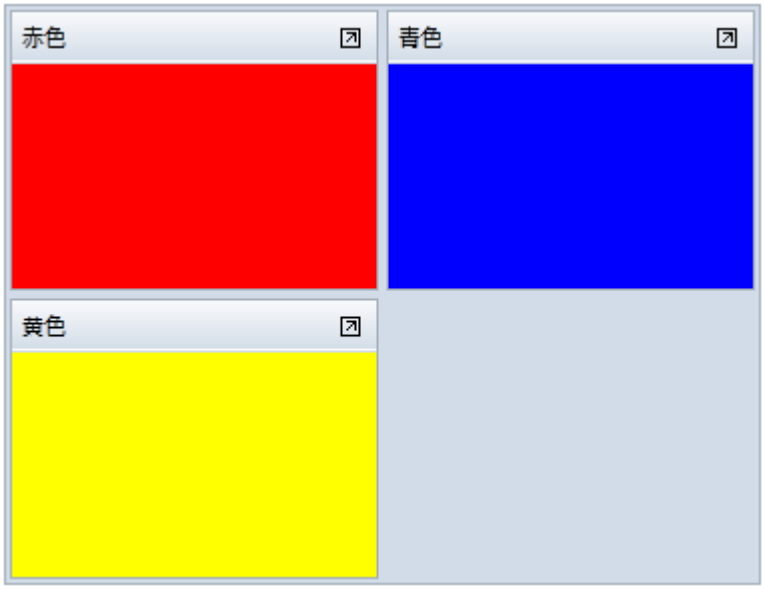
C1TileViewItem コントロールは、ヘッダーとコンテンツ領域の2つの部分で構成されます。次の図に、ツールバーとコンテンツ領域を示します。



C1TileViewItem に追加されたコンテンツは、コンテンツ領域に表示されます。ヘッダー領域には、キャプションバータイトルを追加できます。右上隅のボタンは、**C1TileView** コントロールを最大化または最小化します。

TileViewItem の状態

各 **C1TileViewItem** には、最小化状態、最大化状態、デフォルト状態（最小化状態でも最大化状態でもない）の3つの状態があります。たとえば、次の図では、**C1TileView** コントロール内にある3つの **C1TileViewItem** がすべてデフォルト状態で表示されています。



次の図では、赤色の C1TileViewItem が最大化され、他の2つの項目が最小化されています。

1つの項目が最大化されると、他の項目は最小化され、**MinimizedItem** プロパティで指定された方法で表示されます。デフォルト状態では、Columns プロパティと Rows プロパティを使用してレイアウトが決定されます。最小化/最大化状態では、**MinimizedItemPosition** プロパティを使用してレイアウトが決定されます。

列と行

Columns プロパティと**Rows** プロパティはそれぞれ、**C1TileViewItem** が配置されている列の数と行の数を取得または設定します。この値が0の場合は、スクロールの必要がない最小数が使用されます。**Columns** と **Rows** の両方が0の場合、項目は1つのマス目に配置されます。

デフォルト状態では、**Columns** プロパティと **Rows** プロパティを使用してレイアウトが決定されます。最小化/最大化状態では、MinimizedItemPosition プロパティを使用してレイアウトが決定されます。状態の詳細については、「TileViewItem の状態」を参照してください。

最小化項目の位置

MinimizedItemsPosition プロパティを使用すると、最小化された項目を C1TileView コントロール内のどこに表示するかを指定できます。オプションには**[左]**、**[右]**、**[上]**および**[下]**があります。デフォルトでは、最小化された項目はパネルの右に表示されます。

C1TileView のデフォルト状態では、Columns プロパティと Rows プロパティを使用してレイアウトが決定されます。最小化/最大化状態では、**MinimizedItemPosition** プロパティを使用してレイアウトが決定されます。状態の詳細については、「TileViewItem の状態」を参照してください。

ドラッグアンドドロップ操作

CanUserReorder プロパティを設定して、**C1TileView** コントロールでドラッグアンドドロップ操作を許可するかどうかを簡単に指定できます。デフォルトでは、このプロパティが **True** に設定されており、ユーザーは実行時に項目を並べ替えることができます。反対に、このプロパティが **False** に設定されている場合は、実行時に項目を並べ替えることができません。例については、「ドラッグアンドドロップ機能を無効にする」を参照してください。

基本的なプロパティ

TileView for WPF/Silverlight には、コントロールの機能を設定するためのいくつかのプロパティがあります。主要なプロパティを次に示します。外観を制御するプロパティの詳細については、「外観プロパティ」を参照してください。

次のプロパティを使用して、**C1TileView** コントロールをカスタマイズできます。

プロパティ(WPF)	プロパティ(Silverlight)	説明
AnimationDuration	**AnimationDuration**	項目の並べ替えにかかる時間を取得または設定します。
CanUserReorder	**CanUserReorder**	ユーザーがこのコントロールから**C1TileViewItem** をドラッグアンドドロップして並べ替えできるかどうかを取得または設定します。
Columns	**Columns**	C1TileViewItem が配置されている列の数を取得または設定します。この値が0の場合は、スクロールの必要がない最小数が使用されます。**Columns** と **Rows** の両方が0の場合、項目は1つのマス目に配置されます。
ItemTemplateHeader	**ItemTemplateHeader**	項目のタイトルとして使用される **DataTemplate** を取得または設定します。
ItemTemplateMaximized	**ItemTemplateMaximized**	**Maximized()** 状態の項目に使用される **DataTemplate** を取得または設定します。
ItemTemplateMinimized	**ItemTemplateMinimized**	**Minimized()** 状態の項目に使用される **DataTemplate** を取得または設定します。
MaximizedIndex	**MaximizedIndex**	選択されている項目の **Items** コレクション内のインデックスを取得または設定します。
MaximizedItem	**MaximizedItem**	現在強調表示されている **Items** コレクションのメンバを取得または設定します。
MinimizedItemsPosition	**MinimizedItemsPosition**	項目が最小化された状態のストリップを配置する場所を取得または設定します。スクロールバーは、ストリップの右または下にあります。
Rows	**Rows**	**C1TileViewItem** が配置されている行の数を取得または設定します。この値が0の場合は、スクロールの必要がない最小数が使用されます。**Columns** と **Rows** の両方が0の場合、項目は1つのマス目に配置されます。
ScrollBarStyle	**ScrollBarStyle**	内部スクロールバーに使用されるスタイルを取得または設定します。
ScrollBarVisibility	**ScrollBarVisibility**	スクロールバーを可視にするかどうかを取得または設定します。
UpdateSourceCollection	**UpdateSourceCollection**	項目の順序の変更を **Items** または **ItemsSource** に書き込むかどうかを取得または設定します。

レイアウトおよび外観

以下のトピックでは、**C1TileView** コントロールのレイアウトと外観をカスタマイズする方法について詳しく説明します。組み込みのレイアウトオプションを使用して、グリッドやキャンバスなどのコントロールをパネル内でレイアウトできます。テーマを使用することで、グリッドの外観をカスタマイズしたり、WPF/Silverlight の XAML ベースのスタイル設定を活用することができます。また、テンプレートを使用して、コントロールを書式設定およびレイアウトしたり、コントロールの操作をカスタマイズすることもできます。

パネル内のレイアウト

WPF/Silverlight アプリケーションでは、付属するレイアウトプロパティを使用して、**C1TileView** や他のコントロールを簡単にレイアウトできます。たとえば、**Grid** パネルでは **Row**、**ColumnSpan**、および **RowSpan** プロパティ、**Canvas** パネルでは **Left** および **Top** プロパティを使用して、コントロールをレイアウトできます。たとえば、**Grid** パネル内に配置された**C1TileView** コントロールには、次の **Layout** プロパティがあります。

Grid パネル内で、C1TileView コントロールのサイズ、配置、および場所を変更できます。

外観プロパティ

TileView for WPF/Silverlight には、コントロールの外観をカスタマイズするためのいくつかのプロパティがあります。コントロールの色、境界、および高さを変更できます。以下のトピックでは、これらの外観プロパティの一部について説明します。

色のプロパティ

次のプロパティを使用して、コントロール自体に使用される色をカスタマイズできます。

プロパティ(WPF)	プロパティ (Silverlight)	説明
Background		コントロールの背景を描画するブラシを取得または設定します。これは依存プロパティです。

ButtonBackground	**ButtonBackground**	コントロール内のボタンの背景に割り当てられる **Brush** を取得または設定します。
ButtonForeground	**ButtonForeground**	コントロール内のボタンの前景に割り当てられる **Brush** を取得または設定します。
FocusBrush	**FocusBrush**	フォーカスがあるコントロールを強調表示するために使用される Brush を取得または設定します。
Foreground		前景色を描画するブラシを取得または設定します。これは依存プロパティです。
HeaderForeground	**HeaderForeground**	保持されている**C1TileViewItem** のヘッダーの前景として使用される **Brush** を取得または設定します
ItemBackground	**ItemBackground**	内部のC1TileViewItem の背景として使用される **Brush** を取得または設定します。
ItemForeground	**ItemForeground**	内部のC1TileViewItem の前景として使用される **Brush** を取得または設定します。
MouseOverBrush	**MouseOverBrush**	マウスがあるコントロールを強調表示するために使用される **Brush** を取得または設定します。
PressedBrush	**PressedBrush**	クリックされたボタンを描画するために使用される **Brush** を取得または設定します。

配置プロパティ

次のプロパティを使用して、コントロールの配置をカスタマイズできます。

プロパティ	説明
HorizontalAlignment	パネルや項目コントロールなどの親要素内に置かれる要素に適用される水平配置の特性を取得または設定します。これは依存プロパティです。
VerticalAlignment	パネルや項目コントロールなどの親要素内に置かれる要素に適用される垂直配置の特性を取得または設定します。これは依存プロパティです。

境界線のプロパティ

次のプロパティを使用して、コントロールの境界線をカスタマイズできます。

プロパティ	説明
BorderBrush	コントロールの境界線の背景を描画するブラシを取得または設定します。これは依存プロパティです。
BorderThickness	コントロールの境界線の太さを取得または設定します。これは依存プロパティです。

サイズのプロパティ

次のプロパティを使用すると、コントロールのサイズをカスタマイズできます。

プロパティ	説明
Height	要素の推奨高さを取得または設定します。これは依存プロパティです。

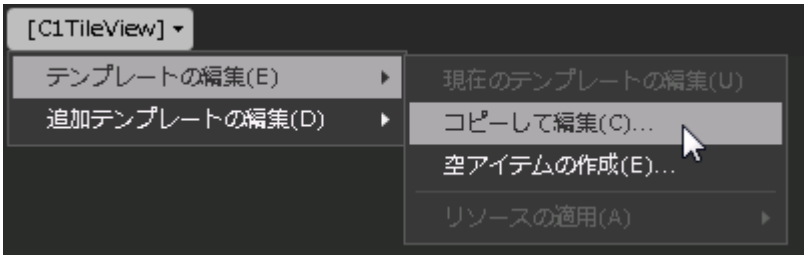
MaxHeight	要素の最大高さ制約を取得または設定します。これは依存プロパティです
MaxWidth	要素の最大幅制約を取得または設定します。これは依存プロパティです。
MinHeight	要素の最小高さ制約を取得または設定します。これは依存プロパティです。
MinWidth	要素の最小幅制約を取得または設定します。これは依存プロパティです。
Width	要素の幅を取得または設定します。これは依存プロパティです。

テンプレート

WPF/Silverlight コントロールを使用する主な利点の1つは、これが自由にカスタマイズできるユーザーインターフェイスを持つ「外観のない」コントロールであることです。WPF/Silverlight アプリケーションのユーザーインターフェイスであるルックアンドフィールを独自に設計するのと同様に、**TileView for WPF/Silverlight** で管理されるデータに関して独自の UI を提供できます。Extensible Application Markup Language（XAML。「ザムル」と発音する）は、コードを記述することなく独自の UI を設計するための簡単な方法を提供する XML ベースの宣言型言語です。

テンプレートへのアクセス

テンプレートにアクセスするには、Microsoft Expression Blend で、**C1TileView** コントロールを選択し、メニューから**[テンプレートの編集]**を選択します。**[コピーして編集]**を選択して現在のテンプレートのコピーを作成して編集するか、**[空アイテムの作成]**を選択して新しい空のテンプレートを作成します。



新しく作成されたテンプレートは、**[オブジェクトとタイムライン]**ウィンドウに表示されます。Template プロパティを使用してテンプレートをカスタマイズできます。

メモ：メニューを使用して新しいテンプレートを作成する場合、テンプレートはそのテンプレートのプロパティに自動的にリンクされます。手作業でテンプレートの XAML を作成する場合は、作成したテンプレートに適切なTemplateプロパティをリンクする必要があります。

追加のテンプレート

デフォルトテンプレートのほかに、C1TileView コントロールには追加のテンプレートがいくつかあります。これらの追加テンプレートには、Microsoft Expression Blend からもアクセスできます。Blend でC1TileView コントロールを選択し、メニューから**[追加テンプレートの編集]**を選択します。テンプレートを選択し、**[空アイテムの作成]**を選択します。

スタイル

TileView for WPF/Silverlight の**C1TileView** コントロールは、コントロールの外観を変更するために使用できるスタイルのプロパティを提供します。これらのスタイルの一部について、次の表で説明します。

スタイル	説明
FocusVisualStyle	この要素がキーボードフォーカスを受け取ったときに適用される外観、効果などのスタイル特性をカスタマイズするためのプロパティを取得または設定します。これは依存プロパティです。
FontStyle	フォントスタイルを取得または設定します。これは依存プロパティです。
ScrollBarStyle(WPF) **ScrollBarStyle**(Sliverlight)	スクロールバーのスタイルを決定します。
Style	この要素のレンダリング時に使用されるスタイルを取得または設定します。これは依存プロパティです。

次の表で、これらのテンプレートの一部について説明します。

スタイル(WPF)	スタイル(Sliverlight)	説明
ItemTemplateHeader	**ItemTemplateHeader**	項目のタイトルとして使用される DataTemplate を取得または設定します。
ItemTemplateMaximized	**ItemTemplateMaximized**	Maximized() 状態の項目に使用される DataTemplate を取得または設定します。
ItemTemplateMinimized	**ItemTemplateMinimized**	Minimized() 状態の項目に使用される DataTemplate を取得または設定します。

表示状態

Microsoft Expression Blend で、カスタム状態や状態グループを追加して、ユーザーコントロールの状態ごとに異なる外観を定義できます。たとえば、マウスが置かれたときのコントロールの表示状態を変更できます。 新しいテンプレートを作成し、新しいテンプレートパーツを追加することで、表示状態を表示および編集できます。これで、そのパーツで利用可能な表示状態が**[表示状態]**ウィンドウに表示されます。

よく使用される状態としては、項目の通常の外観を示す **Normal**、マウスが置かれている項目を示す **MouseOver**、有効でない項目を示す **Disabled** などがあります。フォーカスの状態には、項目にフォーカスがないときの **Unfocused**、項目にフォーカスがあるときの **Focused** などがあります。

タスク別ヘルプ

次のタスク別ヘルプトピックは、ユーザーの皆様が Visual Studio および Expression Blend に精通しており、**C1TileView** コントロールの一般的な使用方法を理解していることを前提としています。**TileView for WPF/Silverlight** 製品に精通していない場合は、まず「クイックスタート」を参照してください。

このセクションの各トピックは、**TileView for WPF/Silverlight** 製品を使用して特定のタスクを実行するためのソリューションを提供します。また、タスク別ヘルプトピックのほとんどは、新しい WPF/Silverlight プロジェクトが作成され、**C1TileView** コントロールがプロジェクトに追加されていることを前提としています（コントロールの作成については、「C1TileView を追加する」を参照）。

C1TileView を追加する

このトピックでは、アプリケーションに**C1TileView** コントロールを追加します。次の手順に従います。

1. Visual Studio の**[ファイル]**メニューから、**[新規作成]**を選択し、**[プロジェクト]**を選択します。
2. **[新しいプロジェクト]**ダイアログボックスで、左側のメニューから言語を選択し、**[フレームワーク]**ドロップダウンリストで**[.NET Framework 4]**を選択し、プロジェクトの名前を入力します。
3. ソリューションエクスプローラで、プロジェクト名を右クリックし、[**参照の追加**]を選択します。[**参照の追加**]ダイアログボックスで、以下のアセンブリを見つけて選択し、**[OK]**をクリックしてプロジェクトに参照を追加します。
    - **C1.WPF** および **C1.Silverlight**
    - **C1.WPF.TileView** および **C1.Silverlight.Tileview**
4. MainWindow.xaml ファイルの XAML ビューを開き、マークアップ を使用して、UserControl タグに XAML 名前空間を追加します。名前空間は次のようになります。

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

WPF

XAML
```
<Window x:Class="MainWindow"
    xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
    xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
    xmlns:c1="http://schemas.componentone.com/winfx/2006/xaml"
    Title="MainWindow" Height="350" Width="525">
```

Silverlight

XAML
```
<UserControl x:Class="QuickStart.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"
xmlns:c1="http://schemas.componentone.com/winfx/2006/xaml" mc:Ignorable="d"
```

```
d:DesignHeight="300" d:DesignWidth="400"><Window x:Class="MainWindow"
    xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
    xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
    xmlns:c1="http://schemas.componentone.com/winfx/2006/xaml"
    Title="MainWindow" Height="350">
```

5. ページの Grid タグ内に <c1:C1TileView x:Name="C1TileView1" /> タグを追加して、アプリケーションに**C1TileView** コントロールを追加します。

XAML

```
<Grid x:Name="LayoutRoot" Background="White">
    <c1:C1TileView x:Name="C1TileView1" />
</Grid>
```

XAML は次のようになります。

これで、"C1TileView1" という名前のC1TileView コントロールがアプリケーションに追加されます。

ここまでの成果

これで、アプリケーションのユーザーインターフェイスが正しくセットアップされましたが、このアプリケーションを実行すると、**C1TileView** コントロールにコンテンツがないことがわかります。詳細については、「項目を追加する」トピックを参照してください。

メモ: C1TileView コントロールが Visual Studio のツールボックスにインストールされている場合は、ページにコントロールをドラッグするだけで、上のすべての手順が自動的に実行されます。

項目を追加する

このトピックでは、**C1TileView** コントロールに**C1TileViewItem** を追加します。このトピックは、空の **C1TileView** コントロールがアプリケーションに追加されていることを前提としています。

<c1:C1TileView x:Name="C1TileView1" /> タグを編集して、いくつかの C1TileViewItem を追加します。XAML は次のようになります。

XAML

```
<c1:C1TileView Name="C1TileView1">
    <c1:C1TileViewItem Background="Red" Header="赤色"></c1:C1TileViewItem>
    <c1:C1TileViewItem Background="Orange" Header="オレンジ色"></c1:C1TileViewItem>
    <c1:C1TileViewItem Background="Yellow" Header="黄色"></c1:C1TileViewItem>
    <c1:C1TileViewItem Background="Green" Header="緑色"></c1:C1TileViewItem>
    <c1:C1TileViewItem Background="Blue" Header="青色"></c1:C1TileViewItem>
    <c1:C1TileViewItem Background="Purple" Header="紫色"></c1:C1TileViewItem>
</c1:C1TileView>
```

これで、**C1TileView** コントロールに6つの**C1TileViewItem** が追加されました。

ドラッグアンドドロップ機能を無効にする

デフォルトでは、ドラッグアンドドロップ機能が有効化されており、ユーザーは実行時に**C1TileViewItem** 要素を並べ替えることができます。ただし、必要に応じて、**CanUserReorder** プロパティを **False** に設定することで、ドラッグアンドドロップ機能を無効にすることができます。

設計時

設計時に[**プロパティ**]ウィンドウで**C1TileView** コントロールでドラッグアンドドロップ機能を無効にするには、次の手順に従います。

1. **C1TileView** コントロールをクリックして選択します。
2. **[プロパティ]**ウィンドウに移動し、**CanUserReorder** プロパティを見つけます。
3. **CanUserReorder** プロパティの横にあるドロップダウン矢印をクリックし、**False** を選択します。これで、ドラッグアンドドロップ機能が無効になります。

XAML の場合

XAML でC1TileView コントロールのドラッグアンドドロップ機能を無効にするには、CanUserReorder="False" を タグに追加します。次のようになります。

XAML

```
<c1:C1TileView Name="C1TileView1" CanUserReorder="False">
```

コードの場合

ウィンドウを右クリックし、**[コードの表示]**を選択してコードエディタを開きます。コードをメインクラス **Window1_Loaded** イベントハンドラに追加します。次のようになります。

VisualBasic

```
Window1_Loaded(ByVal sender As System.Object, ByVal e As
System.Windows.RoutedEventArgs) Handles MyBase.LoadedPublic Sub New()
InitializeComponent()
    Me.C1TileView1.CanUserReorder = False
End Sub
```

C#

```
private void Window_Loaded(object sender, RoutedEventArgs e)public MainPage(){
InitializeComponent();
{
    this.c1TileView1.CanUserReorder = false;
}
```

プロジェクトの実行と確認

実行時にドラッグアンドドロップ操作を実行できなくなります。

ヘッダの外観をカスタマイズする

C1TileView には、C1TileViewItem の **Header** の外観を変更できるプロパティがいくつかあります。これらのプロパティは、Header、HeaderBackground、HeaderFontFamily、HeaderFontSize、HeaderFontStretch、HeaderFontStyle、HeaderFontWeight、HeaderForeground、HeaderPadding、および HeaderTemplate です。

たとえば、次のマークアップは、これらのプロパティのいくつかを設定します。

XAML

```
<c1:C1TileViewItem Header="ニュース" HeaderPadding="10 5 5 5"
HeaderForeground="#FF507494" HeaderFontFamily="Trebuchet MS" HeaderFontSize="16">
   <c1:C1TileViewItem.HeaderBackground>
      <LinearGradientBrush EndPoint="0.5,1" StartPoint="0.5,0">
        <GradientStop Color="#FFE9ECF0" Offset="0" />
            <GradientStop Color="#FFDDE1E7" Offset="0.2" />
            <GradientStop Color="#FFCCD3DC" Offset="0.2" />
            <GradientStop Color="#FFFAFAFB" Offset="0.647" />
      </LinearGradientBrush>
    </c1:C1TileViewItem.HeaderBackground>
</c1:C1TileViewItem>
```

最大化/最小化項目テンプレートを作成する

C1TileView コントロールの項目を最小化および最大化した場合の表示方法をカスタマイズできます。たとえば、最小化された項目のコンテンツ領域にアイコンを表示して、その項目に含まれるコンテンツの種類を示すことができます。**ContentMinimized** プロパティおよび **ContentMaximized** プロパティを使用して、表示テンプレートを設定できます。これらのプロパティが設定されていない場合は、**Content** が使用されます。

たとえば、次のマークアップは、**ContentMinimized** テンプレートと **ContentMaximized** テンプレートを追加します。

XAML

```
<c1:C1TileView Name="C1TileView1">
    <c1:C1TileViewItem Background="Red" Header="赤色">
        <c1:C1TileViewItem.ContentMinimized >
            <Label Content="最小化" Height="28" Name="Label1" Foreground="White"/>
        </c1:C1TileViewItem.ContentMinimized>
        <c1:C1TileViewItem.ContentMaximized >
            <Label Content="最大化" Height="28" Name="Label2" Foreground="White"/>
        </c1:C1TileViewItem.ContentMaximized>
    </c1:C1TileViewItem>
    <c1:C1TileViewItem Background="Orange" Header="オレンジ色">
        <c1:C1TileViewItem.ContentMinimized >
            <Label Content="最小化" Height="28" Name="Label3" Foreground="White"/>
        </c1:C1TileViewItem.ContentMinimized>
        <c1:C1TileViewItem.ContentMaximized >
            <Label Content="最大化" Height="28" Name="Label4" Foreground="White"/>
        </c1:C1TileViewItem.ContentMaximized>
    </c1:C1TileViewItem>
</c1:C1TileView>
```

ここまでの成果

C1TileView の最小化および最大化された状態に使用されるテンプレートを追加しました。アプリケーションを実行し、項目の1つを最大化して、最小化された項目も最大化された項目もコンテンツが変更されることを確認してください。最小化された項目を最大化すると、各項目のコンテンツが再度変更されます。