

# AppView for ASP.NET Web Forms

2018.04.18 更新

グレースィティ株式会社

## 目次

<a href="#">製品の概要</a>	3
<a href="#">ComponentOne for ASP.NET Web Forms のヘルプ</a>	3
<a href="#">主な特長</a>	4
<a href="#">クイックスタート</a>	5
<a href="#">手順 1: ページへの C1AppView の追加</a>	5
<a href="#">手順 2: C1AppViewItems および C1AppViewPages の追加</a>	5-7
<a href="#">手順 3: アプリケーションの実行</a>	7-8
<a href="#">デザイン時のサポート</a>	9
<a href="#">C1AppView スマートタグ</a>	9-10
<a href="#">C1AppView のコンテキストメニュー</a>	10
<a href="#">C1AppView デザイナフォーム</a>	10-11
<a href="#">C1AppView デザイナフォームの操作</a>	11-15
<a href="#">C1AppView デザイナフォームメニュー</a>	15-16
<a href="#">C1AppView デザイナフォームツールバー</a>	16
<a href="#">C1AppView デザイナフォームの使用法</a>	16-17
<a href="#">C1AppViewItem の削除</a>	17
<a href="#">C1AppViewItem の名前の変更</a>	17
<a href="#">子項目の追加</a>	17
<a href="#">C1AppViewItem の挿入</a>	17-18
<a href="#">C1AppView の要素</a>	19
<a href="#">C1AppView</a>	19
<a href="#">コンテンツページ</a>	19-20
<a href="#">ヘッダー要素</a>	20
<a href="#">C1AppViewItem</a>	20
<a href="#">C1AppViewPage</a>	20-21
<a href="#">C1AppView の外観</a>	22
<a href="#">テーマスウォッチ</a>	22
<a href="#">スウォッチ a</a>	22-23
<a href="#">スウォッチ b</a>	23
<a href="#">タスク別ヘルプ</a>	24
<a href="#">テーマ</a>	24

<a href="#">デフォルトのスウォッチの適用</a>	24
<a href="#">コードでのデフォルトテーマの設定</a>	24-25
<a href="#">カスタムテーマの使用</a>	25
<a href="#">チュートリアル</a>	26
<a href="#">イベント計画アプリケーションの作成</a>	26
<a href="#">手順 1: アプリケーションの設定</a>	26-27
<a href="#">手順 2: アプリケーションの Web フォームの作成</a>	27
<a href="#">a. Main.aspx ファイルへのマークアップの追加</a>	27-28
<a href="#">b. カレンダーの作成</a>	28-29
<a href="#">c. イベントの作成</a>	29-37
<a href="#">d. Home フォルダ要素の作成</a>	37-38
<a href="#">手順 3: モデルの作成</a>	38-40
<a href="#">手順 4: アプリケーションの実行</a>	40-43
<a href="#">クライアント側の操作</a>	44
<a href="#">クライアント側イベント</a>	44

## 製品の概要

**AppView for ASP.NET Web Forms** は、インタラクティブなアプリケーションの作成に最適です。このコントロールは、アダプティブなオールインワンアプリケーションを作成するための堅牢なフレームワークを提供できるように特別に作成されています。組み込みのナビゲーション機能、AJAX でロードされるコンテンツ、適合型のレイアウトシステムを備えた単一のアプリケーションを構築できます。このレイアウトシステムは、画面領域を最適化しつつ、スマートフォンとタブレットの両方で機能します。

## ComponentOne for ASP.NET Web Forms のヘルプ

ComponentOne for ASP.NET Web Forms の各コントロールで共通したトピック、アセンブリの追加、テーマの適用、クライアント側情報などについては「[ASP.NET Web Forms ユーザーガイド](#)」を参照してください。

## 主な特長

C1AppView には、次のようなユニークな機能があります。

- **モバイルアプリケーションの最適化**

C1AppView は、特にモバイルアプリケーションの作成を目的として設計されています。このコントロールを使用して、柔軟性が高くタッチ操作に適したモバイルアプリケーションを作成することができます。

- **モバイルアプリケーションに適応**

C1AppView コントロールは、画面サイズに最適なレイアウトを自動的に決定します。タブレットに表示すると、コンテナが左右に分かれて表示され、左側にメニュー、右側にコンテンツペインが表示されます。スマートフォンを使用してアプリケーションを表示すると、最初にメニューが表示されます。メニュー項目を選択すると、画面がコンテンツペインに移動します。

- **単一のアプリケーションの構築**

C1AppViewの堅牢なフレームワークを使用して、ナビゲーション機能が組み込まれたオールインワンアプリケーションを作成できます。

## クイックスタート

このクイックスタートでは、**AppView for ASP.NET Web Forms** を初めて使用するための手順について説明します。このクイックスタートでは、Visual Studio でプロジェクトを作成し、**C1AppView** コントロールをプロジェクトに追加し、**C1AppView** コントロールのコンテンツを追加してカスタマイズします。

## 手順 1: ページへの C1AppView の追加

<>この手順では、新しいプロジェクトを作成し、そのプロジェクトに **AppView for ASP.NET Web Forms** コントロールを追加します。クイックスタートを開始するには、次の手順を実行します。

1. Visual Studio の **[ファイル]** メニューから、**[新規作成]** → **[プロジェクト]** を選択します。**[新しいプロジェクト]** ダイアログボックスが表示されます。
  - a. 左ペインで言語を展開し、**[Web]** を選択します。
  - b. 右ペインで、**[ASP.NET 空の Web アプリケーション]** を選択し、アプリケーションの **名前** を入力して、**[OK]** を選択します。
  - c. 新しいアプリケーションが作成されます。
2. ソリューションエクスプローラで、プロジェクトを右クリックし、**[参照の追加]** を選択します。
  - a. **[参照の追加]** ダイアログボックスで、**C1.Web.Wijmo.Controls.4** アセンブリと **C1.Web.Wijmo.Controls.Design.4** アセンブリを見つけて選択します。
  - b. **[OK]** をクリックしてアセンブリ参照を追加します。
3. ソリューションエクスプローラで、プロジェクトを右クリックし、コンテキストメニューから **[追加]** → **[新しい項目]** を選択します
  - a. **[新しい項目の追加]** ダイアログボックスで、テンプレートのリストから **[Web フォーム]** を選択します。
  - b. この項目名を "Main.aspx" と指定します。
  - c. **[追加]** をクリックして新しいページを開きます。
4. ページ内の `<body>` タグに移動します。マークアップは次の例のようになります

ソースビュー

```
<body>
  <form id="form1" runat="server">
    <div>

  </div>
</form>
</body>
```

5. 既存の `<div>` `</div>` タグを次のように編集します

ソースビュー

```
<div data-role="page"> </div>
```

6. `<div>` `</div>` タグの間にカーソルを置き、以下のいずれかの方法で **C1AppView** コントロールを追加します
  - a. Visual Studio のツールボックスで、**C1AppView** コントロールを見つけます。コントロールをダブルクリックしてアプリケーションに追加します。
  - b. 次のマークアップを `<div>` `</div>` タグの間に挿入します。

ソースビュー

```
<cc1:C1AppView ID="C1AppView1" runat="server" Height="300px" ></cc1:C1AppView>
```

これで、新しいプロジェクトが作成され、プロジェクトに **C1AppView** コントロールが追加されました。次の手順では、このコントロールの外観と動作をカスタマイズします。

## 手順 2: C1AppViewItems および C1AppViewPages の追加

この手順では、アプリケーションに1つの **C1AppViewItem** といくつかの **C1AppViewPage** を追加します。**AppView for ASP.NET Web Forms** コントロールをカスタマイズするには、次の手順を実行します。

1. <cc1:C1AppView> 開始タグをクリックして、コントロールの**C1AppView スマートタグ**  を表示しますが作成されません。
2. **[C1AppView のタスク]メニュー**で、**[モバイルモード]**オプションのチェックボックスをオンにします。
3. ページをクリックして**[C1AppView のタスク]メニュー**を閉じます。
4. 次のように、<cc1:C1AppView> マークアップを変更します。このマークアップは、プロジェクトに1つの**C1AppViewItem** を追加します。

## ソースビュー

```
<cc1:C1AppView ID="C1AppView1" runat="server" HeaderTitle="ホームページ"
Height="300px">
  <DefaultContent>
    <p>
      このアプリケーションは Wijmo とjQuery Mobile を使用してビルドします。
      Wijmo の詳細については、<a href="http://wijmo.com">http://wijmo.com</a> を参
      照してください。
    </p>
  </DefaultContent>
  <Items>
    <cc1:C1AppViewItem Text="Wizard" AppViewPageUrl="/Wizard/Index.aspx">
    </cc1:C1AppViewItem>
  </Items>
</cc1:C1AppView>
```

**AppViewPageUrl** プロパティの設定に注目してください。次の手順では、C1AppViewPage を追加します。

5. ソリューションエクスプローラで、アプリケーションの名前を右クリックします。メニューから**[追加]**→**[新規フォルダ]**を選択します。追加したフォルダ名を **Wizard** と指定します。
6. **Wizard** フォルダを右クリックし、次の手順を実行します。
  - a. メニューから**[追加]**→**[新しい項目]**を選択します。
  - b. **[新しい項目の追加]**ダイアログウィンドウから、**[Web フォーム]**を選択し、**Index.aspx** という名前を指定します。**[OK]**をクリックすると、ページが開きます。
7. **Index.aspx** ページを選択し、<body> タグ内のマークアップを次のように編集します。

## マークアップ

```
<cc1:C1AppViewPage runat="server">
  <content id="Content1" runat="server">
    <Template>
      <cc1:C1Wizard ID="C1Wizard1" runat="server" Delay="300">
        <steps>
          <cc1:C1WizardStep runat="server" Title="Step1"
          ID="C1Wizard1_Step1">
            Mauris eleifend est et turpis. Duis id erat. Duis cursus.
          </cc1:C1WizardStep>
          <cc1:C1WizardStep runat="server" Title="Step2"
          ID="C1Wizard1_Step2">
            This is the second step.
          </cc1:C1WizardStep>
          <cc1:C1WizardStep runat="server" Title="Step3"
          ID="C1Wizard1_Step3">
            This is the third step.
          </cc1:C1WizardStep>
          <cc1:C1WizardStep runat="server" Title="Step4"
          ID="C1Wizard1_Step4">
```

```
ID="C1Wizard1_Step4">
    This is the fourth step.
</cc1:C1WizardStep>
<cc1:C1WizardStep runat="server" Title="Step5"
ID="C1Wizard1_Step5">
    This is the final step.
</cc1:C1WizardStep>
</steps>
</cc1:C1Wizard>
</Template>
</content>
</cc1:C1AppViewPage>
```

これで、新しいプロジェクトが作成され、プロジェクトに **C1AppView** コントロールが追加されました。次の手順では、このコントロールの外観と動作をカスタマイズします。

## 手順 3: アプリケーションの実行

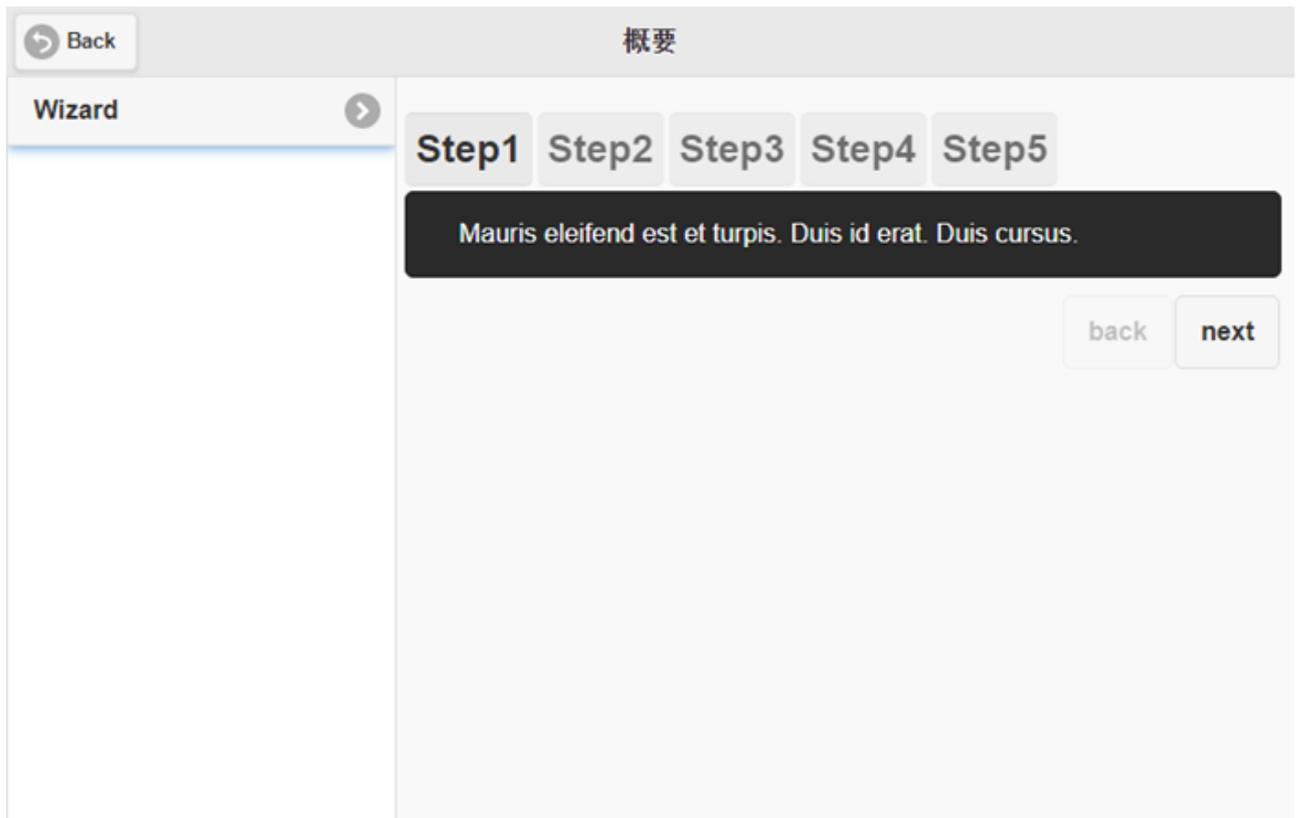
**C1AppView**コントロールの外観と動作をカスタマイズしたので、次はアプリケーションを実行し、**C1AppView**コントロールのいくつかの実行時機能を確認します。

次の手順を実行します。

1. [F5]キーを押すか、デバッグを開始して、アプリケーションを実行します。
2. 最初、アプリケーションは次の図のように表示されます。



3. **[Wizard]**の**C1AppViewItem**をタップまたはクリックすると、**C1Wizard** コントロールを含むページが表示されます。



おめでとうございます。ここでは、[C1AppView](#) コントロールを作成してカスタマイズしました。これで、**AppView for ASP.NET Web Forms** クイックスタートガイドは終了です。

## デザイン時のサポート

**AppView for ASP.NET Web Forms** は、オブジェクトモデルを簡単に利用できるように、カスタマイズされたコンテキストメニュー、スマートタグ、および設計時サポートを備えています。

以下のセクションでは、**AppView for ASP.NET Web Forms** 設計時環境 (特に [C1AppView スマートタグ](#) と [C1AppView のコンテキストメニュー](#)) からアクセスできる **[C1AppView のタスク]** メニュー) を使用して、**C1AppView** コントロールを設定する方法について説明します。

## C1AppView スマートタグ

**C1AppView** コントロールはスマートタグ  を備えています。スマートタグは、各コンポーネント/コマンドで最もよく使用されるプロパティを提供するショートカットタスクメニューです。

**[C1AppView のタスク]** メニューにアクセスするには、**C1AppView** コントロールの右上隅にあるスマートタグの矢印をクリックします。**[C1AppView のタスク]** メニューが表示されます。



**[C1AppView のタスク]** メニューの機能は次のとおりです。

- **AppView 項目の編集**  
[AppView 項目の編集] をクリックすると、**C1AppView デザイナフォーム** が開きます。このフォームを使用して、**C1AppView** のプロパティを設定し、**C1AppViewItem** を追加し、**C1AppViewItem** のプロパティを設定できます。**C1AppView** アプリケーションをプレビューすることもできます。詳細については、「[C1AppView デザイナフォーム](#)」トピックを参照してください。
- **テーマ**  
[テーマ] ドロップダウンボックスをクリックすると、さまざまなビジュアルスキームを選択できます。使用可能なビジュアルスタイルの詳細については、「[C1AppView の外観](#)」を参照してください。
- **新しいテーマの作成**  
[新しいテーマの作成] オプションをクリックすると、**ThemeRoller for Visual Studio** が開きます。アプリケーションで **ThemeRoller for Visual Studio** を使用する方法については、「[ThemeRoller for Visual Studio](#)」を参照してください。
- **CDN の使用**  
[CDN の使用] チェックボックスをオンにして、ウィジェットエクステンダがコンテンツ配信ネットワークからクライアントリソースをロードする必要があることを指定します。このボックスは、デフォルトではオフになっています。
- **CDN パス**  
コンテンツ配信ネットワークのパスを指定します。パスを変更するには、ここに URL を入力します。
- **Bootstrap の使用**  
[Bootstrap の使用] オプションは、Bootstrap 統合を使用するかどうかを指定します。アプリケーションで Bootstrap

テーマを使用する方法については、「[Bootstrap for ASP.NET Web Forms クイックスタート](#)」を参照してください。

- **モバイルモード**  
[C1AppView](#) コントロールと [C1ListView](#) コントロールを使用するには、**[モバイルモード]**をオンにする必要があります。
- **ThemeSwatch**  
[\[ThemeSwatch\]](#) オプションを使用すると、アプリケーション全体のテーマを簡単に作成できます。
- **バージョン情報**  
[\[バージョン情報\]](#) 項目をクリックすると、[\[バージョン情報\]](#) ダイアログボックスが表示され、**AppView for ASP.NET Web Forms** のバージョン番号およびオンラインリソースが表示されます。
- **エクステンダの追加**  
[\[エクステンダの追加\]](#) 項目をクリックすると、[\[エクステンダウィザード\]](#) が開き、[C1AppView](#) コントロールにエクステンダを追加できます。

## C1AppView のコンテキストメニュー

Visual Studio がすべての .NET コントロールに対して提供するコンテキストメニューに、[C1AppView](#) コントロールのコマンドが追加されます。[C1AppView](#) コンテキストメニューにアクセスするには、[C1AppView](#) コントロール内の任意の場所で右クリックします。[C1AppView](#) コンテキストメニューが表示されます。

✂	切り取り(I)	Ctrl+X
📄	コピー(Y)	Ctrl+C
📄	貼り付け(P)	Ctrl+V
	代替の貼り付け(E)	
✖	削除(D)	Del
	コードの表示(C)	
🌐	ブラウザーで表示 (Internet Explorer)(B)	Ctrl+Shift+W
	スマート タグの表示(G)	Shift+Alt+F10
	AppView項目の編集	
🔄	最新の情報に更新(E)	
🔧	プロパティ(R)	Alt+Enter

[C1AppView](#) コンテキストメニューには、[C1AppView](#)によって次のカスタムコマンドが追加されます。

- **AppView 項目の編集**  
[\[AppView 項目の編集\]](#) をクリックすると、[C1AppView デザイナフォーム](#)が開きます。このフォームを使用して、[C1AppView](#)のプロパティを設定し、[C1AppViewItem](#) を追加し、[C1AppViewItem](#)のプロパティを設定できます。[C1AppView](#)アプリケーションをプレビューすることもできます。詳細については、「[C1AppView デザイナフォーム](#)」トピックを参照してください。
- **エクステンダの追加**  
[\[エクステンダの追加\]](#) 項目をクリックすると、[\[エクステンダウィザード\]](#) が開き、[C1AppView](#) コントロールにエクステンダを追加できます。

## C1AppView デザイナフォーム

[C1AppView デザイナフォーム](#)は、プロパティの編集、アプリケーションへの [C1AppViewItems](#) の追加、[C1AppViewItem](#) プロパティの編集、および [C1AppView](#)アプリケーションのプレビューを行うための[C1AppView](#) のデザイナーです。[C1AppView デザイナフォーム](#)は、ユーザーがコントロールを視覚的に変更できるという点で、Visual Studio の[プロパティ]ウィンドウに似ています。ただし、デザイナーフォームでは、[C1AppViewItem](#) の追加、[C1AppView](#)コントロールと [C1AppViewItem](#) コントロールのプロパティの編集、[\[プレビュー\]](#)タブでのアプリケーションのプレビューも行うことができます。

# AppView for ASP.NET WebForms

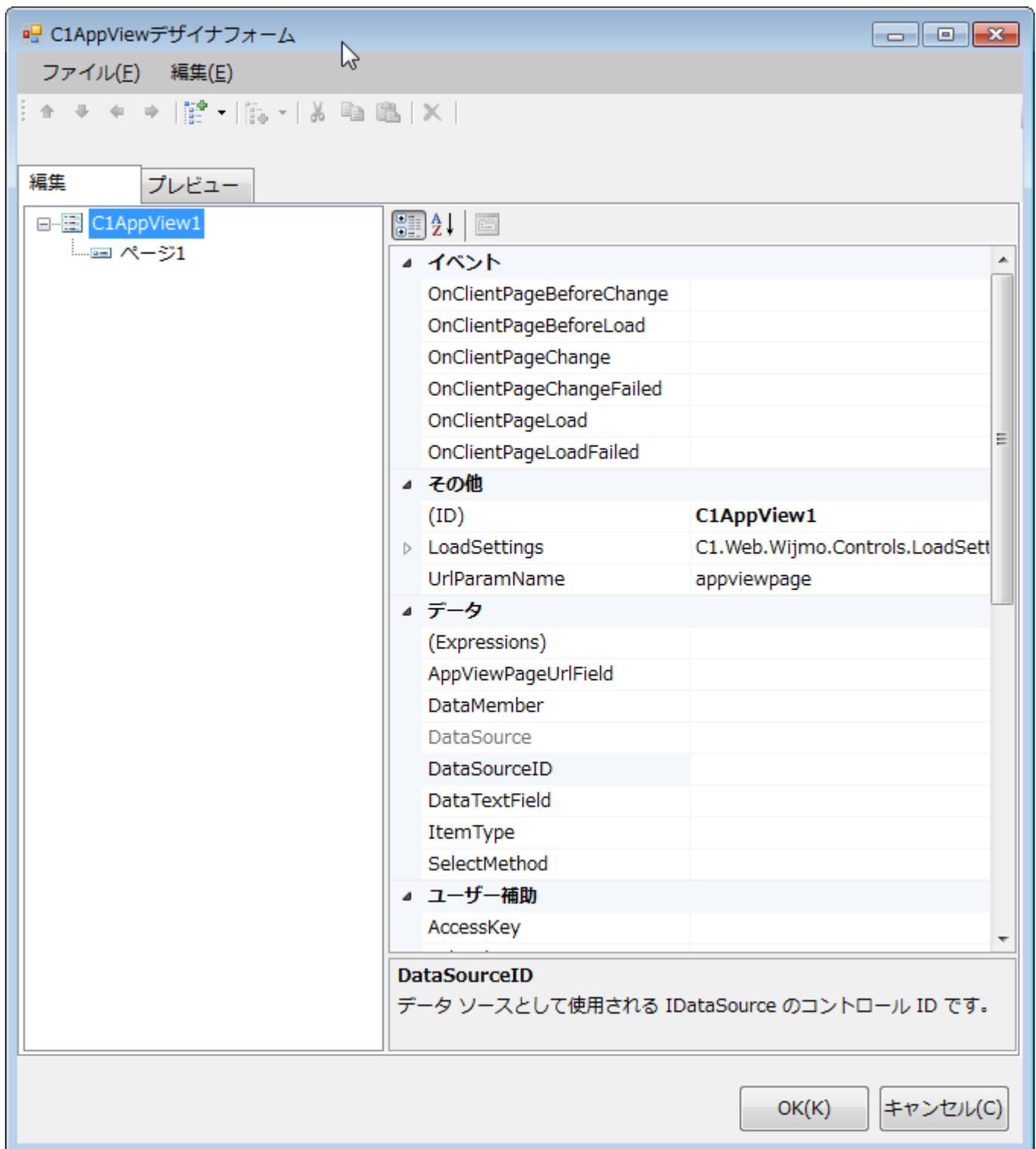
このトピックでは、デザイナフォーム内のコマンドを使用して最小限の作業と時間で **C1AppView** コントロールを編集できるように、**C1AppView デザイナフォーム** のインターフェースについて説明します。

**C1AppView デザイナフォーム** を開くには、**C1AppView スマートタグ** をクリックします。**[C1AppView のタスク]** メニューから **[AppView 項目の編集]** リンクを選択します。



## C1AppView デザイナフォームの操作

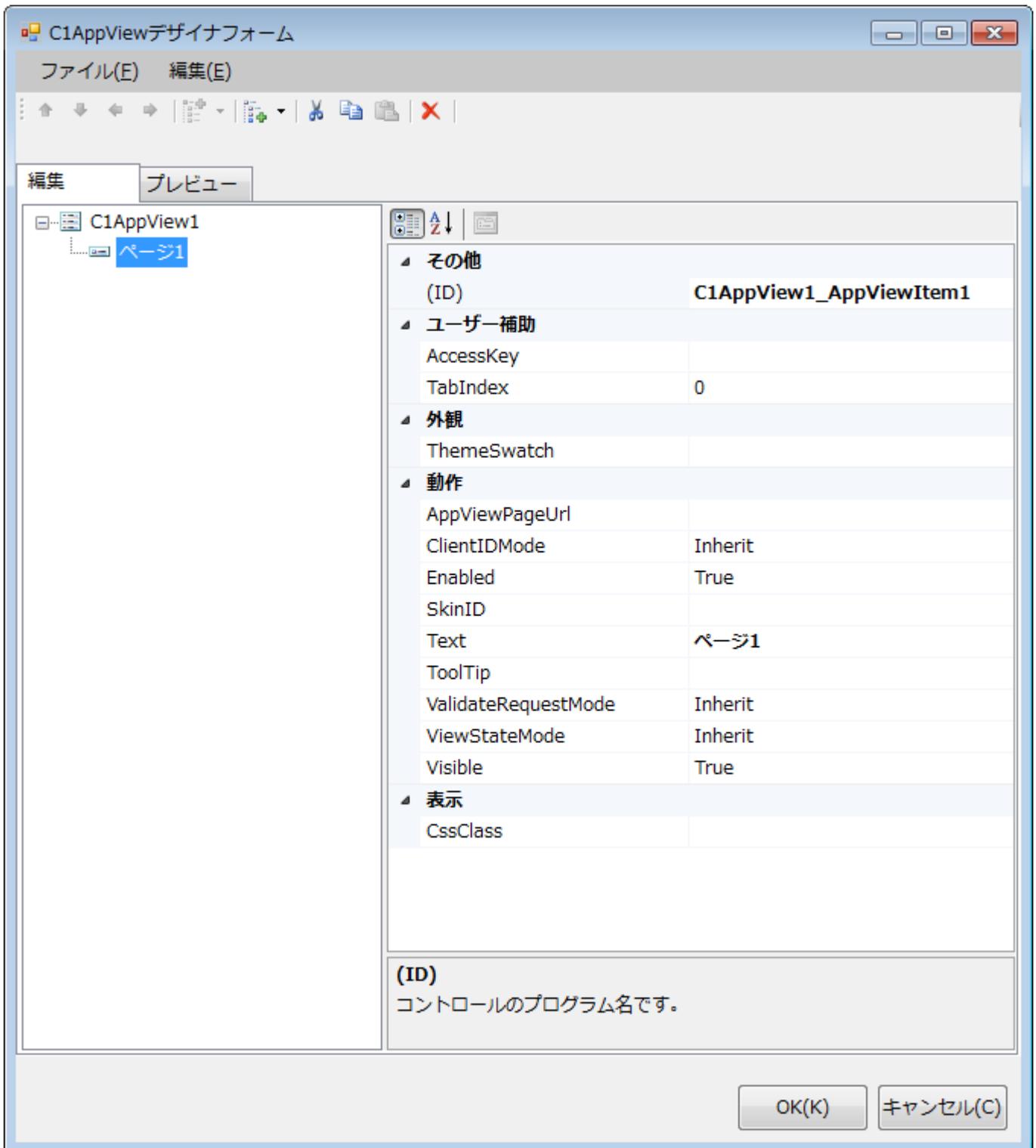
**C1AppView デザイナフォーム** には、メニュー、ツールバー、**[編集]** タブ、および **[プレビュー]** タブがあります。



## [編集]タブ

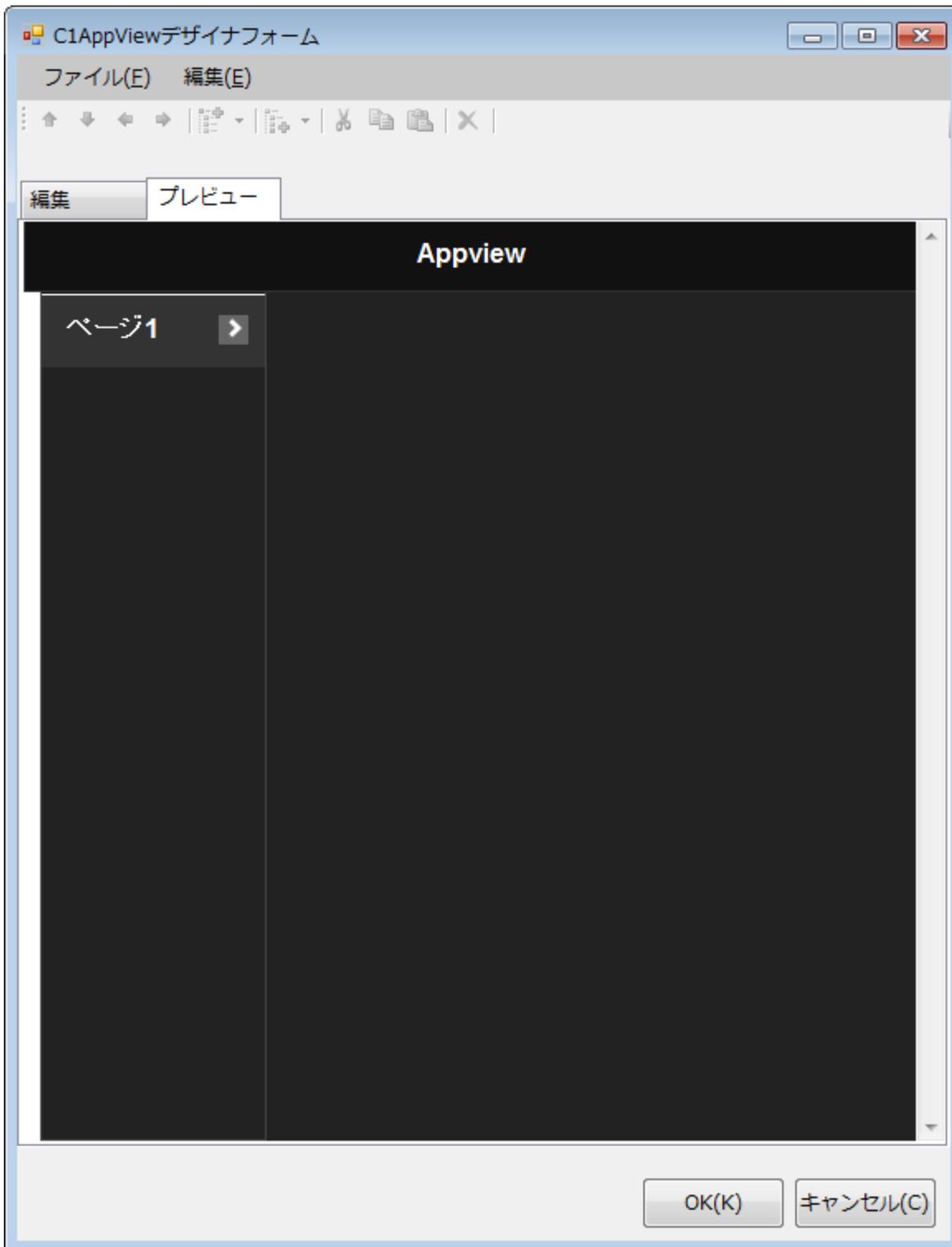
[編集]タブをクリックし、編集する**C1AppView** または **C1AppViewItem** を選択します。これで、編集可能なプロパティが表示されます。

# AppView for ASP.NET WebForms



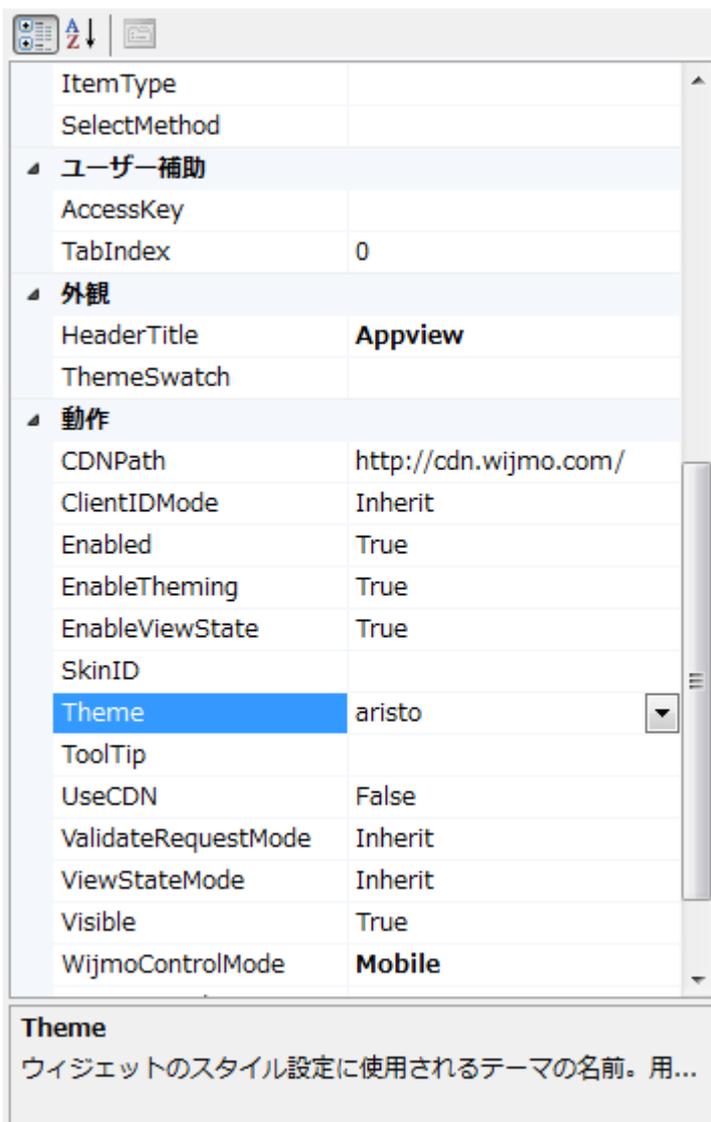
## [プレビュー]タブ

**C1AppView** コントロールをプレビューするには、[プレビュー]タブをクリックします。



## プロパティペイン

C1AppView デザイナフォームのプロパティペインは、Visual Studio の[プロパティ]ウィンドウとほとんど同じです。プロパティペインを使用して、**C1AppView** または**C1AppViewItem**のプロパティを変更します。



## コマンドボタン

次の表は、2つのコマンドボタンの説明です。

ボタン	説明
OK	[OK]をクリックすると、 <b>C1AppView</b> コントロールに新しい設定が適用されます。
Cancel	[キャンセル]をクリックすると、C1AppView デザイナフォームが閉じられます。新しい設定はキャンセルされ、 <b>C1AppView</b> コントロールにはデフォルトの設定が適用されます。

## C1AppView デザイナフォームメニュー

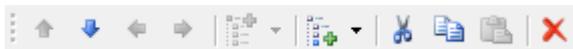
C1AppView デザイナフォームメニューには、以下のメニュー項目とサブメニューがあります。

メニュー項目	サブメニュー項目	サブメニュー項目のアイコン	説明
ファイル	XML からロード		C1AppView コントロールの書式設定を .xml ファイルからロードします。

	XMLとして保存		C1AppView コントロールの現在の書式設定を .xml ファイルに保存します。
	終了		C1AppView デザイナフォームを終了します。
編集	項目の挿入		新しい C1AppViewItem を C1AppViewItem のリスト内の指定された位置に挿入します。
	子の追加		新しい C1AppViewItem を C1AppView または別の C1AppViewItem の子として追加します。
	切り取り		選択された C1AppViewItem を C1AppViewItem のリストから切り取ります。
	コピー		選択された C1AppViewItem をコピーします。
	貼り付け		C1AppViewItem を C1AppViewItem のリスト内の指定された位置に貼り付けます。
	削除		選択された C1AppViewItem を削除します。
	名前の変更		C1AppViewItem の名前を変更できます。

## C1AppView デザイナフォームツールバー

C1AppView デザイナフォームツールバーは、メインのC1AppView コントロールとC1AppViewItemsのどちらを選択しているかによって使用できるオプションが異なります。



**メモ:** [項目を左に移動]および[項目を右に移動]ボタンは使用できません。[項目を上に移動]、[項目を下に移動]、[切り取り]、[コピー]、[貼り付け]、および[削除]ボタンは、C1AppViewItem を選択している場合にのみ使用できます。

次の表で、使用可能なボタンについて説明します。

ボタン	名前	説明
	項目を上に移動	選択された C1AppViewItem を項目リストの1つ上に移動します。このボタンは、C1AppViewItem を選択した場合に使用できます。
	項目を下に移動	選択された C1AppViewItem > を項目リストの1つ下に移動します。このボタンは、C1AppViewItem を選択した場合に使用できます。
	子項目の追加	C1AppView コントロールに C1AppViewItem を追加します。このボタンは、C1AppView コントロールを選択している場合にのみ使用できます。
	項目の挿入	選択された C1AppViewItem の上にC1AppViewItem挿入します。このボタンは、C1AppViewItem を選択した場合にのみ使用できます。
	切り取り	選択された C1AppViewItem をC1AppViewItemのリストから切り取ります。
	コピー	選択された C1AppViewItem をコピーします。
	貼り付け	C1AppViewItem を C1AppViewItemのリスト内の指定された位置に貼り付けます。
	削除	選択された C1AppViewItem を削除します。

## C1AppView デザイナフォームの使用方法

以下のトピックでは、**C1AppView デザイナフォーム**を使用していくつかのタスクを実行する方法について説明します。

### C1AppViewItem の削除

C1AppViewItem を削除するには、以下のいずれかの方法を使用できます。

1. **ショートカットメニューを使用して C1AppViewItem を削除する**  
削除する **C1AppViewItem** を右クリックし、ショートカットメニューから[削除]を選択します。
2. **[削除]ボタンを押して C1AppViewItem を削除する**  
削除する **C1AppViewItem** を選択し、[削除]ボタンを押します。
3. **[編集]メニューを使用して C1AppViewItem を削除する**  
削除する **C1AppViewItem** を選択します。[編集]メニューから[削除]を選択します。

### C1AppViewItem の名前の変更

C1AppViewItem の名前を変更するには、以下のいずれかの方法を使用できます。

- **[F2]キーを押す**
  1. 名前を変更する **C1AppViewItem** を選択します。
  2. [F2]キーを押し、**C1AppViewItem** の新しい名前を入力します。
- **ショートカットメニューから[名前の変更]を選択する**
  1. 名前を変更する **C1AppViewItem**を選択します。
  2. **C1AppViewItem**の新しい名前を入力します。
- **[編集]メニューから[名前の変更]を選択する**
  1. 名前を変更する **C1AppViewItem**を選択します。
  2. **[編集]**→**[名前の変更]**を選択し、**C1AppViewItem**の新しい名前を入力します。
- **プロパティペインで C1AppViewItem の名前を変更する**
  1. 名前を変更する **C1AppViewItem**を選択します。プロパティペインにプロパティが表示されます。
  2. **Text** プロパティを見つけます。[Text]プロパティ領域に **C1AppViewItem**の新しい名前を入力します。

### 子項目の追加

C1AppView コントロールに子の **C1AppViewItem**を追加するには、以下のいずれかの方法を使用できます。

- **ショートカットメニューを使用して子項目を追加する**  
C1AppView を右クリックし、ショートカットメニューから[子の追加]→[AppViewItem]を選択します。
- **[子項目の追加]ボタンを押して子項目を追加する**  
**C1AppViewItem**の追加先の C1AppView を選択します。デザイナフォームツールバーにある[子項目の追加]ボタンをクリックすると、**C1AppViewItem**が追加されます。
- **[編集]メニューを使用して子項目を追加する**  
**C1AppViewItem**の追加先の C1AppView を選択します。[編集]メニューから[子の追加]→[AppView 項目]を選択します。

### C1AppViewItem の挿入

C1AppViewItemを挿入するには、以下のいずれかの方法を使用できます。

- **ショートカットメニューを使用して C1AppViewItem を挿入する**
  1. C1AppViewItem を右クリックします。
  2. ショートカットメニューから[項目の挿入]→[AppView 項目]を選択します。
  3. 選択した項目の上に、C1AppViewItemが追加されます。
- **[項目の挿入]ボタンを使用して C1AppViewItem を挿入する**

C1AppViewItemを選択します。[項目の挿入]ボタンを押すと、選択されている項目の上に、もう1つのC1AppViewItemが挿入されます。
- **[編集]メニューを使用して C1AppViewItem を挿入する**

C1AppViewItemの追加先の C1AppViewItemを選択します。[編集]メニューから[項目の挿入]→[AppView 項目]を選択します。

## C1AppView の要素

このセクションでは、**C1AppView** コントロールを構成する要素について画像を使用してわかりやすく説明します。このコントロールの UI には、コンテンツページ、いくつかの **C1AppViewItem** で構成されたメニュー要素、およびページヘッダー要素が含まれます。スマートフォンなどの小型のモバイルデバイスでは、メニュー要素が表示されます。次の画像は、ラベルを付けた **C1AppView** コントロールです。



## C1AppView

**C1AppView** コントロールは、メニュー要素、コンテンツページ、ヘッダー要素の3つで構成されます。メニュー要素の詳細については、「**C1AppViewItem**」トピックを参照してください。C1AppView コントロールを作成するには、Visual Studio のツールボックスでこのコントロールを見つけ、ダブルクリックしてアプリケーションに追加するか、アプリケーションの タグ内に次のマークアップを置きます。

### ソースビュー

```
<cc1:C1AppView ID="C1AppView1" runat="server"> </cc1:C1AppView>
```

**メモ:** **C1AppView** コントロールが正しく機能するには、C1AppView コントロールをモバイルモードに設定する必要があります。このプロパティは、[C1AppView のタスク]メニューを使用して設定できます。

以下のトピックでは、**C1AppView** コントロールを構成する要素と、それらの要素の作成に使用するマークアップについて説明します。メニュー要素および要素の作成に使用するマークアップの詳細については、「**C1AppViewItem**」トピックを参照してください。

## コンテンツページ

コンテンツページは、C1AppView アプリケーションの右側に表示されます。コンテンツページには、<DefaultContent> タグ内で設定したコンテンツが表示されます。次の例は、コンテンツを設定するためのマークアップです。

## ソースビュー

```
<cc1:C1AppView ID="C1AppView1" runat="server" WijmoControlMode="Mobile">
  <DefaultContent>
    <p>
      このアプリケーションは Wijmo と jQuery Mobile を使用してビルドします。Wijmo の詳細については、<a
      href="http://wijmo.com">http://wijmo.com</a> を参照してください。
    </p>
  </DefaultContent>
</cc1:C1AppView>
```

コンテンツページには、他のコントロールを始めとする任意のコンテンツを表示できます。

## ヘッダー要素

ヘッダー要素を使用して、アプリケーションにタイトルを追加できます。ヘッダー要素は、アプリケーションの上部にある横長のヘッダーバーとして表示されます。ヘッダーは、次のマークアップに示すように、**HeaderTitle** プロパティを使用して設定できます。

## ソースビュー

```
<cc1:C1AppView ID="C1AppView1" runat="server" HeaderTitle="Home Page" WijmoControlMode="Mobile">
```

## C1AppViewItem

**C1AppViewItem**を使用して、メニュー要素を作成します。メニュー要素は、**C1AppView** コントロールの左側に表示されま  
す。次のマークアップに示すように、これらの項目は `<Items>` タグ内に置かれます。

## ソースビュー

```
<cc1:C1AppView ID="C1AppView1" runat="server" HeaderTitle="Home Page" Height="300px"
WijmoControlMode="Mobile">
  <Items>
    <cc1:C1AppViewItem Text="List view" AppViewPageUrl="/Event/Index.aspx"> </cc1:C1AppViewItem>
    <cc1:C1AppViewItem Text="Calendar view" AppViewPageUrl="/Calendar/Index.aspx"> </cc1:C1AppViewItem>
    <cc1:C1AppViewItem Text="About" AppViewPageUrl="/Home/About.aspx"> </cc1:C1AppViewItem>
    <cc1:C1AppViewItem Text="Contact" AppViewPageUrl="/Home/Contact.aspx"> </cc1:C1AppViewItem>
  </Items>
</cc1:C1AppView>
```

**AppViewPageUrl** プロパティは、**C1AppViewItem**の正しい位置を指定します。上の例では、**AppViewPageUrl** プロパティ  
でローカルの **C1AppViewPage** を指定しています。

## C1AppViewPage

**C1AppViewPage** に含まれる項目は、**C1AppView** コントロールのコンテンツページに表示されます。C1AppViewPage の  
作成に使用するマークアップは次のようになります。

## ソースビュー

```
<cc1:C1AppViewPage ID="C1AppViewPage1" runat="server"> </cc1:C1AppViewPage>
```

# AppView for ASP.NET WebForms

このマークアップで作成される **C1AppViewPage** には、コンテンツが何もアタッチされていません。コンテンツを含む **C1AppViewPage** を作成するには、次のようなマークアップを使用する必要があります。

## ソースビュー

```
<cc1:C1AppViewPage ID="C1AppViewPage1" runat="server">
  <Content ID="Content1" runat="server">
    <Template>
      <cc1:C1ListView ID="ListView1" runat="server" Inset="true">
        <Items>
          <cc1:C1ListViewItem Text="Item 1"></cc1:C1ListViewItem>
          <cc1:C1ListViewItem Text="Item 2"></cc1:C1ListViewItem>
          <cc1:C1ListViewItem Text="Item 3"></cc1:C1ListViewItem>
          <cc1:C1ListViewItem Text="Item 4"></cc1:C1ListViewItem>
          <cc1:C1ListViewItem Text="Item 5"></cc1:C1ListViewItem>
          <cc1:C1ListViewItem Text="Item 6"></cc1:C1ListViewItem>
          <cc1:C1ListViewItem Text="Item 7"></cc1:C1ListViewItem>
        </Items>
      </cc1:C1ListView>
    </Template>
  </Content>
</cc1:C1AppViewPage>
```

これで、基本的な表示リストを含む **C1AppViewPage** が作成されます。**C1AppViewPage** には、テキストや他のコントロールを始めとする任意のコンテンツを表示できます。

## C1AppView の外観

C1AppView は、簡単にカスタマイズできるように設計されています。C1AppView の変更の際に、その可能性は無限大です。さらなるカスタマイズのために、C1AppView では jQuery Mobile テーマスウォッチが使用されます。jQuery Mobile Themeroollerを使用して、独自のテーマをデザインすることもできます。

## テーマスウォッチ

C1AppView では、デフォルトテーマとして jQuery Mobile テーマスウォッチが使用されます。デフォルトテーマには2つのスウォッチが用意されています。C1AppView コントロール内でこれらのスウォッチを組み合わせることで、コントロールを完全にカスタマイズできます。

ThemeSwatch プロパティを使用して、C1AppView コントロール全体に1つのスウォッチを適用することも、コントロール内の C1AppViewPage または C1AppViewItem ごとに異なるスウォッチを適用することもできます。デフォルトでは、C1AppView コントロールは次のように表示されます。



以下のトピックでは、各スウォッチを C1AppView コントロールに適用した例を示します。

## スウォッチ a

ホームページ	
リストビュー	➤
カレンダービュー	➤
情報	➤
連絡先	➤
このアプリケーションは Wijmo とjQuery Mobile を使用してビルドします。 Wijmo の詳細については、 <a href="https://www.grapecity.co.jp/developer/wijmo">https://www.grapecity.co.jp/developer/wijmo</a> を参照してください。	

## スウォッチ b

ホームページ	
リストビュー	➤
カレンダービュー	➤
情報	➤
連絡先	➤
このアプリケーションは Wijmo とjQuery Mobile を使用してビルドします。 Wijmo の詳細については、 <a href="https://www.grapecity.co.jp/developer/wijmo">https://www.grapecity.co.jp/developer/wijmo</a> を参照してください。	

## タスク別ヘルプ

タスク別ヘルプセクションは、ユーザーの皆様が Visual Studio ASP.NET 環境でのプログラミングに精通しており、**AppView for ASP.NET Web Forms** 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。

各トピックでは、**C1AppView** コントロールを使用した特定のタスクのソリューションを提供します。各トピックに示される手順に従って作業を進めるだけで、さまざまな **C1AppView** の機能を使用したプロジェクトを作成できます。

また、タスク別ヘルプトピックは、新しい ASP.NET プロジェクトが既に作成されていることを前提としています。

## テーマ

jQuery Mobile テーマを使用すると、モバイルアプリケーションのテーマも簡単に作成できます。デフォルトのテーマスウォッチを組み合わせることで独自のテーマを作成したり、jQuery の新しい **Mobile ThemeRoller** を使用してデザインしたカスタムテーマを適用することができます。

## デフォルトのスウォッチの適用

デフォルトのテーマスウォッチを組み合わせることで、独自のアプリケーションを作成できます。**C1AppView** アプリケーションにデフォルトのスウォッチを適用するには、次の手順に従います。

1. **C1AppView** コントロールのマークアップは次のようになります。

ソースビュー

```
<cc1:C1AppView ID="C1AppView1" runat="server" HeaderTitle="Wizard" Height="300px" >
  <Items>
    <cc1:C1AppViewItem Text="Wizard Page" AppViewPageUrl="~/Wizard/Index.aspx">
  </cc1:C1AppViewItem>
  </Items>
</cc1:C1AppView>
```

2. コントロールのスマートタグをクリックし、[**C1AppView のタスク**]メニューから[**AppView 項目の編集**]を選択して、**C1AppView デザイナフォーム**を開きます。
3. **C1AppView** コントロールを選択し、プロパティペインで Appearance プロパティを見つけます。
4. **ThemeSwatch** プロパティを「b」に設定し、[**OK**]を押します。アプリケーションの `<cc1:C1AppView>` 開始タグは次のサンプルのようになります。

ソースビュー

```
<cc1:C1AppView ID="C1AppView1" runat="server" HeaderTitle="Wizard" Height="300px" ThemeSwatch="e">
```

5. [F5]キーを押すか、デバッグを開始して、アプリケーションを実行します。次の図のようになります。



## コードでのデフォルトテーマの設定

デフォルトテーマスウォッチの1つをコードで設定することもできます。

1. デザインビューまたはソースビューのいずれかでページを右クリックし、リストから[**コードの表示**]を選択して、コードビューに切り替えます。
2. PageLoad イベントに次のコードを挿入します。

ソースビュー

```
<cc1:C1AppView ID="C1AppView1" runat="server" HeaderTitle="Wizard" Height="300px" >
  <Items>
    <cc1:C1AppViewItem Text="Wizard Page" AppViewPageUrl="~/Wizard/Index.aspx">
  </cc1:C1AppViewItem>
  </Items>
</cc1:C1AppView>
```

## カスタムテーマの使用

jQuery Mobile Themerollerを使用すると、独自のテーマを直観的な方法で簡単にデザインできます。それには、カラーピッカーからスウォッチの目的のセクションまで色をドラッグするだけです。新しいテーマをダウンロードして保存したら、次の手順に従ってアプリケーションにテーマを適用します。

1. ソリューションエクスプローラで、[**すべてのファイルを表示**]を選択して、アプリケーションの非表示ファイルを表示します。
2. **Content** フォルダを見つけ、右クリックしてリストを表示します。[**プロジェクトに追加**]を選択します。
3. **Content** フォルダを再度右クリックし、リストから[**追加**]→[**既存の項目**]を選択します。
4. カスタムテーマを含むファイルを参照して選択し、縮小テーマファイルを選択します。[**追加**]をクリックして、ファイルをアプリケーションに追加します。
5. アプリケーションの <head> </head> タグに次の参照を追加します。テーマ名は、独自のテーマに付けた名前を参照するように変更してください。

ソースビュー

```
<cc1:C1AppView ID="C1AppView1" runat="server" HeaderTitle="Wizard" Height="300px" >
  <Items>
    <cc1:C1AppViewItem Text="Wizard Page" AppViewPageUrl="~/Wizard/Index.aspx">
  </cc1:C1AppViewItem>
  </Items>
</cc1:C1AppView>
```

6. アプリケーションで使用するスウォッチを選択し、**ThemeSwatch** プロパティを設定します。マークアップは次のようになります。

ソースビュー

```
<cc1:C1AppView ID="C1AppView1" runat="server" HeaderTitle="Wizard" Height="300px" >
  <Items>
    <cc1:C1AppViewItem Text="Wizard Page" AppViewPageUrl="~/Wizard/Index.aspx">
  </cc1:C1AppViewItem>
  </Items>
</cc1:C1AppView>
```

7. [F5]キーを押すか、デバッグを開始して、アプリケーションを実行します。カスタムテーマがアプリケーションに適用されたことを確認します。

## チュートリアル

このセクションのチュートリアルでは、手順を追って説明を行います。**AppView for ASP.NET Web Forms** の予備知識は特に必要ありません。このセクションの手順に従って作業を進めるだけで、**AppView for ASP.NET Web Forms** のさまざまな機能を具体的に紹介するプロジェクトを作成できます。これにより、AppView for ASP.NET Web Forms を使用して実行する方法を理解できます。

## イベント計画アプリケーションの作成

**C1AppView** コントロールと **C1ListView** コントロールを使用して、やや複雑なイベント計画アプリケーションを簡単に作成できます。

### 手順 1: アプリケーションの設定

この手順ではアプリケーションを設定します。最初に、新しいアプリケーションを作成し、適切なアセンブリへの参照を追加し、このイベントアプリケーションに必要なフォルダとファイルを作成します。

- 新しい空の ASP.NET アプリケーションを作成します。
  - Visual Studio の [ファイル] メニューから **[新規作成]** → **[プロジェクト]** を選択します。
  - C# または Visual Basic の横のドロップダウン矢印を使用して、テンプレートリストを展開し、**[Web]** を選択します。
  - [ASP.NET 空の Web アプリケーション]** を選択し、アプリケーションの名前を入力して、**[OK]** をクリックします。
- [参照]** フォルダを右クリックし、リストから **[参照の追加]** を選択して、ASP.NET Web Forms アセンブリへの参照を追加します。以下の参照アセンブリを参照して見つけます。
  - C1.Web.Wijmo.Controls.4.dll**
  - C1.Web.Wijmo.Controls.Design.4.dll**
  - Newtonsoft.Json.dll**
  - EntityFramework.dll**
- アプリケーション名を右クリックし、リストから **[追加]** → **[Web フォーム]** を選択します。Web フォームの名前を入力し（この場合は、Main）、**[OK]** をクリックします。
- この手順では、アプリケーションに4つのフォルダを追加します。
  - アプリケーション名を右クリックし、リストから **[追加]** → **[新しいフォルダ]** を選択します。フォルダ名を **Calendar** と指定します。
  - アプリケーション名を右クリックし、リストから **[追加]** → **[新しいフォルダ]** を選択します。フォルダ名を **Event** と指定します。
  - アプリケーション名を右クリックし、リストから **[追加]** → **[新しいフォルダ]** を選択します。フォルダ名を **Home** と指定します。
  - アプリケーション名を右クリックし、リストから **[追加]** → **[新しいフォルダ]** を選択します。フォルダ名を **Models** と指定します。
- Calendar** フォルダを右クリックし、リストから **[追加]** → **[Web フォーム]** を選択します。この Web フォーム名を **Index** と指定します。
- この手順では、**Event** フォルダに5つの Web フォームを追加します。
  - Event** フォルダを右クリックし、リストから **[追加]** → **[Web フォーム]** を選択します。Web フォーム名を **Create** と指定します。
  - Event** フォルダを右クリックし、リストから **[追加]** → **[Web フォーム]** を選択します。Web フォーム名を **Delete** と指定します。
  - Event** フォルダを右クリックし、リストから **[追加]** → **[Web フォーム]** を選択します。Web フォーム名を **Details** と指定します。
  - Event** フォルダを右クリックし、リストから **[追加]** → **[Web フォーム]** を選択します。Web フォーム名を **Edit** と指定します。
  - Event** フォルダを右クリックし、リストから **[追加]** → **[Web フォーム]** を選択します。この Web フォーム名を **Index** と指定します。
- この手順では、**Home** フォルダに3つの Web フォームを追加します。

- **Home** フォルダを右クリックし、リストから[追加]→[Web フォーム]を選択します。Web フォーム名を **About** と指定します。
  - **Home** フォルダを右クリックし、リストから[追加]→[Web フォーム]を選択します。Web フォーム名を **Contact** と指定します。
  - **Home** フォルダを右クリックし、リストから[追加]→[Web フォーム]を選択します。この Web フォーム名を **Index** と指定します。
8. この手順では、**Models** フォルダに3つのコードファイルを追加します。
- **Models** フォルダを右クリックし、リストから[追加]→[コードファイル]を選択します。コードファイル名を **Event** と指定します。
  - **Models** フォルダを右クリックし、リストから[追加]→[コードファイル]を選択します。コードファイル名を **EventAction** と指定します。
  - **Models** フォルダを右クリックし、リストから[追加]→[コードファイル]を選択します。コードファイル名を **EventPlannerEntities** と指定します。

この手順では、新しい ASP.NET アプリケーションを作成し、適切なアセンブリ参照を追加し、イベントカレンダーアプリケーションの構造を設定しました。手順2では、作成した Web フォームにマークアップとスクリプトを追加します。

## 手順 2: アプリケーションの Web フォームの作成

次の手順では、前の手順で作成した Web フォームにマークアップとコードを追加します。

### a. Main.aspx ファイルへのマークアップの追加

1. Main.aspx ファイルをダブルクリックして開きます。
2. Page 呼び出しの直下に、次のコードを追加して、C1.Web.Wijmo.Controls.4 アセンブリを登録します。

ソースビュー

```
<%@ Register Assembly="C1.Web.Wijmo.Controls.4"
Namespace="C1.Web.Wijmo.Controls.C1AppView" TagPrefix="cc1" %>
```

3. ページ内の `<form>` `</form>` タグに移動します。このタグの間に次のマークアップを挿入します。

#### 追加するマークアップ

```
<div data-role="page">
  <cc1:C1AppView ID="C1AppView1" runat="server" >
    <LoadSettings Type="Get" LoadMsgDelay="100" />
    <DefaultContent>
      <p>
        このアプリケーションは Wijmo とjQuery Mobile を使用してビルドします。Wijmo の詳細については、<a href="http://wijmo.com">http://wijmo.com</a> を参照してください。
      </p>
    </DefaultContent>
    <Items>
      <cc1:C1AppViewItem Text="Events List" AppViewPageUrl="/Event/Index.aspx">
    </cc1:C1AppViewItem>
      <cc1:C1AppViewItem Text="Calendar view" AppViewPageUrl="/Calendar/Index.aspx">
    </cc1:C1AppViewItem>
      <cc1:C1AppViewItem Text="About" AppViewPageUrl="/Home/About.aspx">
    </cc1:C1AppViewItem>
      <cc1:C1AppViewItem Text="Contact" AppViewPageUrl="/Home/Contact.aspx">
    </cc1:C1AppViewItem>
    </Items>
  </cc1:C1AppView>
</div>
```

```
</cc1:C1AppView>
</div>
```

## b. カレンダーの作成

1. **Calendar** フォルダから、**Index.aspx** ファイルをダブルクリックして開きます。
2. 以下のアセンブリ登録を追加します。

ソースビュー

```
<%@ Register Assembly="C1.Web.Wijmo.Controls.4"
Namespace="C1.Web.Wijmo.Controls.C1EventsCalendar" TagPrefix="cc1" %>

<%@ Register Assembly="C1.Web.Wijmo.Controls.4"
Namespace="C1.Web.Wijmo.Controls.C1AppView" TagPrefix="cc1" %>
```

3. ページ内の `<form>` `</form>` タグに移動します。このタグの間に次のマークアップを挿入します。

### 追加するマークアップ

```
<asp:EntityDataSource ID="EventPlannerEntityDataSource" runat="server"
    OnContextCreating="EventPlannerEntityDataSource_ContextCreating"
    EntitySetName="Events">
</asp:EntityDataSource>
<cc1:C1AppViewPage ID="C1AppViewPage1" runat="server" HeaderTitle="Event Calendar">
    <Header ID="Header1" runat="server">
        <Template>
            <a href=" ../Main.aspx" data-icon="back">Back</a>
            <h2>Event Calendar</h2>
        </Template>
    </Header>
    <Content ID="Content1" runat="server">
        <Template>
            <cc1:C1EventsCalendar ID="C1EventsCalendar1" runat="server">
                <DataStorage>
                    <EventStorage DataSourceID="EventPlannerEntityDataSource">
                        <Mappings>
                            <IdMapping MappingName="Id" />
                            <SubjectMapping MappingName="Subject" />
                            <StartMapping MappingName="Start" />
                            <EndMapping MappingName="End" />
                            <LocationMapping MappingName="Location" />
                            <DescriptionMapping MappingName="Description" />
                        </Mappings>
                    </EventStorage>
                </DataStorage>
            </cc1:C1EventsCalendar>
        </Template>
    </Content>
</cc1:C1AppViewPage>
```

4. ページを右クリックし、リストから[コードの表示]を選択します。コード内で参照する名前空間を確認します。名前空間は次のようになります。Models への参照では、リストの最後の参照のように、アプリケーション名を使用する必要があります。

## C# コードの書き方

```
C#  
  
using System;  
using System.Collections.Generic;  
using System.Data.Entity.Infrastructure;  
using System.Diagnostics;  
using System.Linq;  
using System.Web;  
using System.Web.UI;  
using System.Web.UI.WebControls;  
using YourApplicationName.Models;
```

5. **Page\_Load** イベントの下に次のコードを追加します。

```
ソースビュー  
  
protected void EventPlannerEntityDataSource_ContextCreating(object sender,  
EntityDataSourceContextCreatingEventArgs e)  
{  
    var db = EventAction.GetEventDb();  
    e.Context = (db as IObjectContextAdapter).ObjectContext;  
}  
}
```

## c. イベントの作成

この手順では、アプリケーションのイベントを作成するために使用されるマークアップとコードを追加します。

1. **Event** フォルダから、**Create.aspx** ファイルをダブルクリックして開きます。
  - 次のコードを追加して、以下のアセンブリを登録します。

```
ソースビュー  
  
<%@ Register Assembly="C1.Web.Wijmo.Controls.4"  
Namespace="C1.Web.Wijmo.Controls.C1ListView" TagPrefix="cc1" %>  
  
<%@ Register Assembly="C1.Web.Wijmo.Controls.4"  
Namespace="C1.Web.Wijmo.Controls.C1AppView" TagPrefix="cc1" %>
```

- ページ内の `<body>` `</body>` タグに移動し、このタグの間に次のマークアップを挿入します。

### 追加するマークアップ

```
<asp:EntityDataSource ID="EventPlannerEntityDataSource" runat="server"  
    OnContextCreating="EventPlannerEntityDataSource_ContextCreating"  
    EntitySetName="Events">  
</asp:EntityDataSource>  
<cc1:C1AppViewPage ID="C1AppViewPage1" runat="server" HeaderTitle="Event Calendar">  
    <Header ID="Header1" runat="server">  
        <Template>  
            <a href=" ../Main.aspx" data-icon="back">Back</a>  
            <h2>Event Calendar</h2>  
        </Template>  
    </Header>  
<Content ID="Content1" runat="server">
```

```

<Template>
  <cc1:C1EventsCalendar ID="C1EventsCalendar1" runat="server">
    <DataStorage>
      <EventStorage DataSourceID="EventPlannerEntityDataSource">
        <Mappings>
          <IdMapping MappingName="Id" />
          <SubjectMapping MappingName="Subject" />
          <StartMapping MappingName="Start" />
          <EndMapping MappingName="End" />
          <LocationMapping MappingName="Location" />
          <DescriptionMapping MappingName="Description" />
        </Mappings>
      </EventStorage>
    </DataStorage>
  </cc1:C1EventsCalendar>
</Template>
</Content>
</cc1:C1AppViewPage>

```

- ページを右クリックし、リストから**[コードの表示]**を選択します。参照が次のようになっていることを確認します。

#### C# コードの書き方

C#

```

using System;
using System.Collections.Generic;
using System.IO;
using System.Linq;
using System.Web;
using System.Web.UI;
using System.Web.UI.WebControls;
using YourApplicationName.Models;

```

- **Page\_PreRenderComplete** イベントに次のコードを追加します。

#### C# コードの書き方

C#

```

protected void Page_PreRenderComplete(object sender, EventArgs e)
{
    if (!IsPostBack)
    {
        EventObj newEvent = EventAction.Create();
        SubjectInput.Text = newEvent.Subject;
        LocationInput.Text = newEvent.Location;
        StartInput.Text = newEvent.Start.ToString("G");
        EndInput.Text = newEvent.End.ToString("G");
        DescriptionInput.Text = newEvent.Description;
        AllDayInput.SelectedON = newEvent.AllDay;
        createForm.Action = "/Event/Create.aspx";
    }
    else
    {
        EventObj newEvent = EventAction.Create();
    }
}

```

```
newEvent.Subject = SubjectInput.Text;
newEvent.Location = LocationInput.Text;
newEvent.Start = DateTime.Parse(StartInput.Text);
newEvent.End = DateTime.Parse(EndInput.Text);
newEvent.Description = DescriptionInput.Text;
newEvent.AllDay = AllDayInput.SelectedON;
EventAction.Add(newEvent);
Response.Redirect("../Main.aspx#appviewpage=Event/Index.aspx");
}
}
}
```

## 2. Event フォルダから、Delete.aspxファイルをダブルクリックして開きます。

- 次のコードを追加して、以下のアセンブリを登録します。

### ソースビュー

```
<%@ Register Assembly="C1.Web.Wijmo.Controls.4"
Namespace="C1.Web.Wijmo.Controls.C1ListView" TagPrefix="cc1" %>

<%@ Register Assembly="C1.Web.Wijmo.Controls.4"
Namespace="C1.Web.Wijmo.Controls.C1AppView" TagPrefix="cc1" %>
```

- ページ内の <body> </body> タグに移動し、このタグの間に次のマークアップを挿入します。

### 追加するマークアップ

```
<cc1:C1AppViewPage ID="C1AppViewPage1" runat="server" HeaderTitle="Delete">
  <Header ID="Header1" runat="server">
    <Template>
      <h2>削除</h2>
    </Template>
  </Header>
  <Content ID="Content1" runat="server">
    <Template>
      <form id="deleteForm" runat="server">
        <cc1:C1ListView ID="C1ListView1" runat="server" Inset="true">
          <Items>
            <cc1:C1ListViewInputItem ID="SubjectInput" LabelText="Subject" Type="label">
          </cc1:C1ListViewInputItem>
          </Items>
        </cc1:C1ListView>
        <input type="button" value="Delete" data-theme="e"
onclick="$('#deleteForm').trigger('submit')" />
        <a href="Index.aspx" data-theme="c" data-role="button">Cancel</a>
      </form>
    </Template>
  </Content>
</cc1:C1AppViewPage>
```

- ページを右クリックし、リストから**[コードの表示]**を選択します。参照が次のようになっていることを確認します。

### C# コードの書き方

```
C#
using System;
```

```
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.Web.UI;
using System.Web.UI.WebControls;
using YourApplicationName.Models;
```

- 既存のコードを次のように編集します。

### C# コードの書き方

```
C#
public partial class Delete : System.Web.UI.Page
{
    protected int id;
    protected void Page_PreRenderComplete(object sender, EventArgs e)
    {
        if (!IsPostBack)
        {
            id = int.Parse(Request["id"]);
            EventObj detail = EventAction.GetEventDetail(id);
            SubjectInput.Text = detail.Subject;
            deleteForm.Action = "/Event/Delete.aspx?id=" + id.ToString();
        }
        else
        {
            id = int.Parse(Request["id"]);
            EventAction.Delete(id);
            Response.Redirect("../Main.aspx#appviewpage=Event/Index.aspx");
        }
    }
}
```

3. **Event** フォルダから、**Details** ファイルをダブルクリックして開きます。**Details.aspx** ファイルが開きます。
  - ページの上部で、以下のブロックを使用してアセンブリを登録します。

#### ソースビュー

```
<%@ Register Assembly="C1.Web.Wijmo.Controls.4"
Namespace="C1.Web.Wijmo.Controls.C1ListView" TagPrefix="cc1" %>

<%@ Register Assembly="C1.Web.Wijmo.Controls.4"
Namespace="C1.Web.Wijmo.Controls.C1AppView" TagPrefix="cc1" %>
```

- ページ内の `<body>` `</body>` タグに移動します。このタグの間に次のマークアップを挿入します。

### 追加するマークアップ

```
<form id="detailForm" runat="server">
  <cc1:C1AppViewPage ID="C1AppViewPage1" runat="server" HeaderTitle="Details">
    <Header ID="Header1" runat="server">
      <Template>
        <a href="Index.aspx" data-icon="back">戻る</a>
        <h2>詳細</h2>
        <a href="Edit.aspx?id=<%=id %>" data-icon="gear">Edit</a>
```

```
</Template>
</Header>
<Content ID="Content1" runat="server">
  <Template>
    <cc1:C1ListView ID="C1ListView1" runat="server" Inset="true">
      <Items>
        <cc1:C1ListViewInputItem ID="SubjectInput" LabelText="Subject" Type="label">
</cc1:C1ListViewInputItem>
        <cc1:C1ListViewInputItem ID="LocationInput" LabelText="Location" Type="label">
</cc1:C1ListViewInputItem>
        <cc1:C1ListViewInputItem ID="StartInput" LabelText="Start" Type="label">
</cc1:C1ListViewInputItem>
        <cc1:C1ListViewInputItem ID="EndInput" LabelText="End" Type="label">
</cc1:C1ListViewInputItem>
        <cc1:C1ListViewInputItem ID="DescriptionInput" LabelText="Description"
Type="label"></cc1:C1ListViewInputItem>
        <cc1:C1ListViewFlipSwitchItem ID="AllDayInput" LabelText="AllDay" Disable="true"
ONMessage="Yes" ONValue="true" OFFMessage="No" OFFValue="false">
</cc1:C1ListViewFlipSwitchItem>
      </Items>
    </cc1:C1ListView>
    <a href="Delete.aspx?id=<%=id %>" data-role="button" data-theme="e">Delete</a>
  </Template>
</Content>
</cc1:C1AppViewPage>
</form>
```

- ページを右クリックし、リストから[コードの表示]を選択します。参照が次のようになっていることを確認します。

## C# コードの書き方

```
C#
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.Web.UI;
using System.Web.UI.WebControls;
using YourApplicationName.Models;
```

- 既存のコードを次のように編集します。

## C# コードの書き方

```
C#
public partial class Details : System.Web.UI.Page
{
    protected int id;
    protected void Page_PreRenderComplete(object sender, EventArgs e)
    {
        if (!IsPostBack)
        {
            id = int.Parse(Request["id"]);
            EventObj eventDetail = EventAction.GetEventDetail(id);
        }
    }
}
```

```

        SubjectInput.Text = eventDetail.Subject;
        LocationInput.Text = eventDetail.Location;
        StartInput.Text = eventDetail.Start.ToString("G");
        EndInput.Text = eventDetail.End.ToString("G");
        DescriptionInput.Text = eventDetail.Description;
        AllDayInput.SelectedON = eventDetail.AllDay;
    }
}
}
}

```

#### 4. Event フォルダから、Edit.aspxファイルをダブルクリックして開きます。

- ページ宣言の下に次のブロックを追加して、C1ListView および ソースビュー

```

<%@ Register Assembly="C1.Web.Wijmo.Controls.4"
Namespace="C1.Web.Wijmo.Controls.C1ListView" TagPrefix="cc1" %>
<%@ Register Assembly="C1.Web.Wijmo.Controls.4"
Namespace="C1.Web.Wijmo.Controls.C1AppView" TagPrefix="cc1" %>

```

- ページ内の <body> </body> タグに移動します。このタグの間に次のマークアップを挿入します。

#### 追加するマークアップ

```

<cc1:C1AppViewPage ID="C1AppViewPage1" runat="server" HeaderTitle="Edit">
  <Header ID="Header1" runat="server">
    <Template>
      <a href="Index.aspx">キャンセル</a>
      <h2>作成</h2>
      <input type="button" value="Save" class="ui-btn-right" data-theme="b"
onclick="$('#editForm').trigger('submit')" />
    </Template>
  </Header>
  <Content ID="Content1" runat="server">
    <Template>
      <form id="editForm" runat="server">
        <cc1:C1ListView ID="C1ListView1" runat="server" Inset="true">
          <Items>
            <cc1:C1ListViewInputItem ID="SubjectInput" LabelText="Subject" Type="text">
</cc1:C1ListViewInputItem>
            <cc1:C1ListViewInputItem ID="LocationInput" LabelText="Location" Type="text">
</cc1:C1ListViewInputItem>
            <cc1:C1ListViewInputItem ID="StartInput" LabelText="Start" Type="text">
</cc1:C1ListViewInputItem>
            <cc1:C1ListViewInputItem ID="EndInput" LabelText="End" Type="text">
</cc1:C1ListViewInputItem>
            <cc1:C1ListViewInputItem ID="DescriptionInput" LabelText="Description" Type="text">
</cc1:C1ListViewInputItem>
            <cc1:C1ListViewFlipSwitchItem ID="AllDayInput" LabelText="AllDay"
ONMessage="Yes" ONValue="true" OFFMessage="No" OFFValue="false">
</cc1:C1ListViewFlipSwitchItem>
          </Items>
        </cc1:C1ListView>
      </form>
    </Template>

```

```
</Content>  
</cc1:C1AppViewPage>
```

- ページを右クリックし、リストから[コードの表示]を選択します。参照が次のようになっていることを確認します。

## C# コードの書き方

```
C#  
  
using System;  
using System.Collections.Generic;  
using System.Linq;  
using System.Web;  
using System.Web.UI;  
using System.Web.UI.WebControls;  
using YourApplicationName.Models;
```

- 既存のコードを次のように編集します。

## C# コードの書き方

```
C#  
  
public partial class Edit : System.Web.UI.Page  
{  
    private EventObj detail;  
    protected int id;  
    protected void Page_PreRenderComplete(object sender, EventArgs e)  
    {  
        if (!IsPostBack)  
        {  
            id = int.Parse(Request["id"]);  
            detail = EventAction.GetEventDetail(id);  
            SubjectInput.Text = detail.Subject;  
            LocationInput.Text = detail.Location;  
            StartInput.Text = detail.Start.ToString("G");  
            EndInput.Text = detail.End.ToString("G");  
            DescriptionInput.Text = detail.Description;  
            AllDayInput.SelectedON = detail.AllDay;  
            editForm.Action = "/Event/Edit.aspx?id=" + id.ToString();  
        }  
        else  
        {  
            id = int.Parse(Request["id"]);  
            detail = EventAction.GetEventDetail(id);  
            detail.Subject = SubjectInput.Text;  
            detail.Location = LocationInput.Text;  
            detail.Start = DateTime.Parse(StartInput.Text);  
            detail.End = DateTime.Parse(EndInput.Text);  
            detail.Description = DescriptionInput.Text;  
            detail.AllDay = AllDayInput.SelectedON;  
            EventAction.Edit(detail);  
            Response.Redirect("../Main.aspx#appviewpage=Event/Index.aspx");  
        }  
    }  
}
```

```
}

```

5. **Event** フォルダから、**Index** ファイルをダブルクリックして開きます。**Index.aspx** ファイルが開きます。
- ページ宣言の下に次のブロックを追加して、**C1ListView** および **C1AppView** アセンブリを登録します。

#### ソースビュー

```
<%@ Register Assembly="C1.Web.Wijmo.Controls.4"
Namespace="C1.Web.Wijmo.Controls.C1ListView" TagPrefix="cc1" %>

<%@ Register Assembly="C1.Web.Wijmo.Controls.4"
Namespace="C1.Web.Wijmo.Controls.C1AppView" TagPrefix="cc1" %>

```

- ページ内の `<body>` `</body>` タグに移動します。このタグの間に次のマークアップを挿入します。

#### 追加するマークアップ

```
<form id="form1" runat="server">
  <cc1:C1AppViewPage ID="C1AppViewPage1" runat="server" HeaderTitle="List View">
    <Header ID="Header1" runat="server">
      <Template>
        <a href=" ../Main.aspx" data-icon="back">戻る</a>
        <h2>リストビュー</h2>
        <a href="Create.aspx" data-icon="plus" data-iconpos="notext">Add</a>
      </Template>
    </Header>
    <Content ID="Content1" runat="server">
      <Template>
        <cc1:C1ListView ID="C1ListView1" runat="server" Inset="true">
        </cc1:C1ListView>
      </Template>
    </Content>
  </cc1:C1AppViewPage>
</form>

```

- ページを右クリックし、リストから**[コードの表示]**を選択します。参照が次のようになっていることを確認します。

#### C# コードの書き方

C#

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.Web.UI;
using System.Web.UI.HtmlControls;
using System.Web.UI.WebControls;
using C1.Web.Wijmo.Controls.C1ListView;
using C1.Web.Wijmo.Controls.C1ListView;
using YourApplicationName.Models;

```

- 既存のコードを次のサンプルのように編集します。

#### C# コードの書き方

C#

```
public partial class Index : System.Web.UI.Page

```

```
{
    protected void Page_Load(object sender, EventArgs e)
    {
        if (!IsPostBack)
        {
            foreach (EventObj item in EventAction.GetEvents())
            {
                C1ListViewLinkItem listItem = new C1ListViewLinkItem();
                listItem.Text = item.Subject;
                listItem.NavigateUrl = "/Event/Details.aspx?id=" + item.Id;
                C1ListView1.Items.Add(listItem);
            }
        }
    }
}
```

## d. Home フォルダ要素の作成

この手順では、**Home** フォルダ内のアプリケーション要素を作成します。

1. **Home** フォルダから、**About** ファイルをダブルクリックして開きます。既存のマークアップを次の内容に置き換えます。

### 追加するマークアップ

```
<div data-role="appviewpage" data-title="About">
  <div data-role="header"><h2>バージョン情報</h2></div>
  <div data-role="content">
    <article>
      <p>
        追加情報を示すために、この領域を使用します。
      </p>
    </article>
    <aside>
      <h3>Aside Title</h3>
      <p>
        追加情報を示すために、この領域を使用します。
      </p>
      <ul>
        <li><a href="Index.aspx">Home</a></li>
        <li><a href="About.aspx">About</a></li>
        <li><a href="Contact.aspx">Contact</a></li>
      </ul>
    </aside>
  </div>
</div>
```

2. **Home** フォルダから、**Contact** ファイルをダブルクリックして開きます。既存のマークアップを次の内容に置き換えます。

### 追加するマークアップ

```
<div data-role="appviewpage" data-title="Contact">
  <div data-role="header"><h2>連絡先</h2></div>
  <div data-role="content">
    <h3>電話番号</h3>
    <span>425.555.0100</span>
    <h3>メール</h3>
    <span><a href="mailto:General@example.com">General@example.com</a></span>
  </div>
</div>
```

3. **Home** フォルダから、**Index** ファイルをダブルクリックして開きます。既存のマークアップを次の内容に置き換えます。

#### 追加するマークアップ

```
<div data-role="appviewpage" data-title="Index">
  <div data-role="header"><h2>ホームページ</h2></div>
  <div data-role="content">
    <p>
      このアプリケーションは Wijmo と jQuery Mobile を使用してビルドします。Wijmo の詳細については、
      <a href="http://wijmo.com"> を参照してください。
    </p>
  </div>
</div>
```

## 手順 3: モデルの作成

この手順では、アプリケーションで使用するモデルを作成します。

1. **Event** フォルダから、**Create.aspx** ファイルをダブルクリックして開きます。
  - 参照が次のようになっていることを確認します。

#### C# コードの書き方

```
C#
using System;
using System.Collections.Generic;
using System.Data.Entity;
using System.Linq;
using System.Web;
```

- 名前空間宣言の下に次のコードを追加します。

#### C# コードの書き方

```
C#
public class EventObj
{
  [Key]
  public int Id { get; set; }
  public string Subject { get; set; }
  public string Location { get; set; }
  public DateTime Start { get; set; }
  public DateTime End { get; set; }
```

```
public string Description { get; set; }
public bool AllDay { get; set; }
}
```

2. **Models** フォルダから、再度 **EventAction.cs** ファイルをダブルクリックして開きます。
  - 参照が次のようになっていることを確認します。

## C# コードの書き方

C#

```
using System;
using System.Collections.Generic;
using System.Data.Entity;
using System.Linq;
using System.Web;
```

- 名前空間宣言の下に次のコードを追加します。

## C# コードの書き方

C#

```
public static class EventAction
{
    private static EventPlannerEntities _eventDb = new EventPlannerEntities();
    internal static EventPlannerEntities GetEventDb()
    {
        return new EventPlannerEntities();
    }
    public static IList GetEvents()
    {
        return _eventDb.Events.ToList();
    }
    public static EventObj GetEventDetail(int id)
    {
        return _eventDb.Events.Find(id);
    }
    public static EventObj Create()
    {
        return new EventObj
        {
            Subject = "New event",
            Start = DateTime.Today,
            End = DateTime.Today.AddDays(1).AddSeconds(-1),
            AllDay = false
        };
    }
    public static void Add(EventObj eventObj)
    {
        _eventDb.Events.Add(eventObj);
        _eventDb.SaveChanges();
    }
    public static void Edit(EventObj eventObj)
    {
        _eventDb.Entry(eventObj).State = EntityState.Modified;
    }
}
```

```

        _eventDb.SaveChanges();
    }
    public static void Delete(int id)
    {
        EventObj eventObj = _eventDb.Events.Find(id);
        _eventDb.Events.Remove(eventObj);
        _eventDb.SaveChanges();
    }
}
}
}

```

3. **Models**フォルダ内の最後のコードファイルをダブルクリックして、**EventPlannerEntities.cs** ファイルを開きます。
  - 参照が次のようになっていることを確認します。

#### C# コードの書き方

```

C#
using System;
using System.Collections.Generic;
using System.Data.Entity;
using System.Linq;
using System.Web;

```

- 名前空間宣言の下に次のコードを追加します。

#### C# コードの書き方

```

C#
public class EventPlannerEntities : DbContext
{
    public DbSet Events { get; set; }
    public EventPlannerEntities()
    {
        Database.CreateIfNotExists();
    }
}
}

```

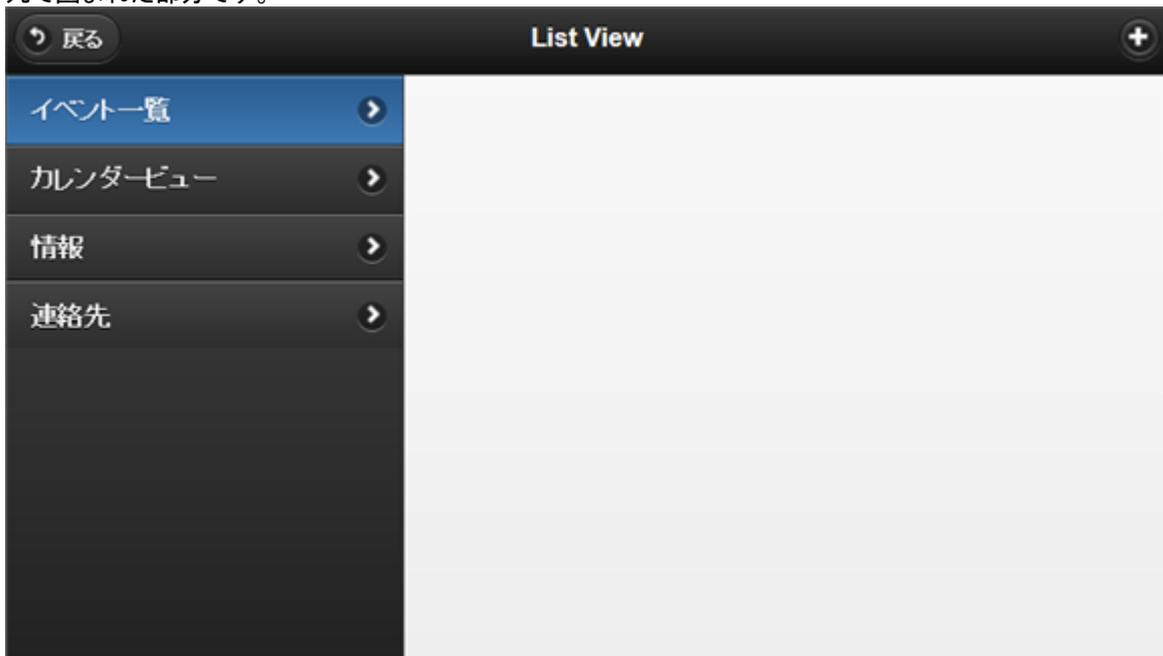
## 手順 4: アプリケーションの実行

この手順ではアプリケーションを実行します。

1. [F5]キーを押すか、デバッグを開始して、アプリケーションを実行します。最初、アプリケーションは次の図のように表示されます。



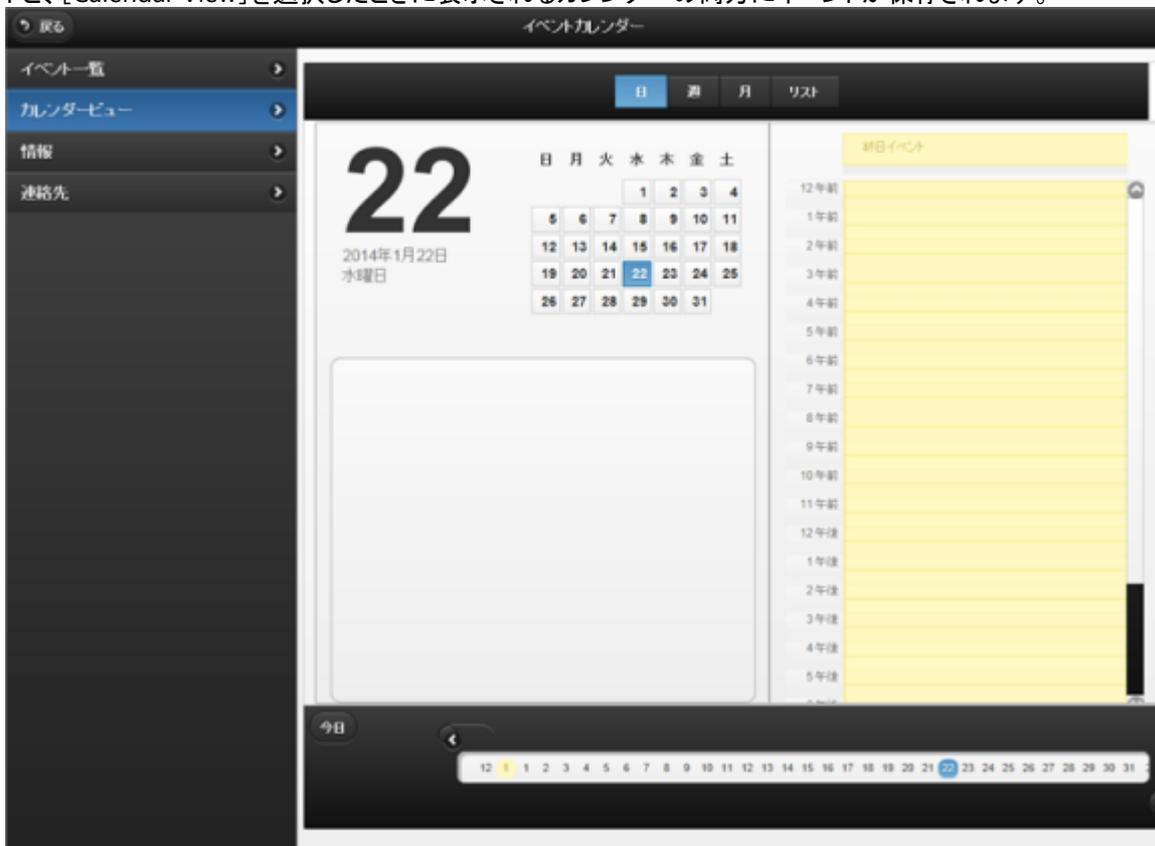
2. [Events List]の C1AppViewItem を初めてクリックまたはタップした際は、何もイベントがリストされません。イベントを作成するには、アプリケーションの右上にある[Add]ボタンをクリックまたはタップします。[Add]ボタンは、次の画像の丸で囲まれた部分です。



3. [Add]ボタンをクリックまたはタップすると、アプリケーションに[Create]フォームがロードされます。



4. イベントフォームに情報を入力し、[Save]をクリックまたはタップすると、[Events List]を選択したときに表示されるリストと、[Calendar View]を選択したときに表示されるカレンダーの両方にイベントが保存されます。



おめでとうございます。これで、イベント計画アプリケーションを作成するチュートリアルは終了です。このチュートリアルで

## AppView for ASP.NET WebForms

は、**C1AppView**、**C1ListView**、および **C1Calendar** コントロールを使用して、イベントを作成、保存、および表示できるアプリケーションを構築しました。このアプリケーションを使用して、リストや **C1Calendar** コントロールにイベントを表示できます。イベントをカレンダーに表示すると、アプリケーションは少し複雑になりますが、イベントが開催される場所や時間を簡単に確認でき、イベントが重なることを避けやすくなります。

## クライアント側の操作

AppView for ASP.NET Web Forms コントロールは、メンバのほとんどがサーバー側コントロールのメンバと同じで、極めて多機能なクライアント側オブジェクトモデルを備えています。

AppView コントロールがレンダリングされると、クライアント側コントロールのインスタンスが自動的に作成されます。すなわち、サーバーにポストバックしなくても、**C1AppView** コントロールのプロパティやメソッドに容易にアクセスできます。

クライアント側コードを使用して、Web ページにさまざまな機能を実装できます。Web サーバーに情報を送信する必要がないため、時間も節約できます。クライアント側オブジェクトモデルを使用することで、Web サイトの効率が高まります。

## クライアント側イベント

**AppView for ASP.NET Web Forms** には、ダイアログウィンドウのサイズ変更などのアクションが発生した際に **C1AppView** コントロールを操作するためのクライアント側イベントが含まれます。

クライアント側イベントの表にリストされているサーバー側のプロパティを使用して、特定のクライアント側イベントに応答する JavaScript 関数の名前を指定できます。たとえば、**C1AppViewPage** が変更されたときに応答する "pagechange" という名前の JavaScript 関数を割り当てるには、**OnClientPageChange** プロパティを「pagechange」に設定します。

次の表に、クライアントスクリプトで使用できるイベントをリストします。これらのプロパティはサーバー側で定義されますが、各 JavaScript 関数に対して宣言する名前の実際のイベントはクライアント側で定義されます。

イベントのサーバー側 プロパティ名	イベント名	説明
<b>OnClientPageBeforeChanged</b>	pagebeforechanged	ページが変更される前に発生します。
<b>OnClientPageBeforeLoad</b>	pagebeforeload	ページがロードされる前発生します。
<b>OnClientPageChange</b>	pagechange	ページが変更されたときに発生します。
<b>OnClientPageChangeFailed</b>	pagechangefailed	ページの変更が失敗したときに発生します。
<b>OnClientPageLoad</b>	pageload	ページがロードされたときに発生します。
<b>OnClientPageLoadFailed</b>	pageloadfailed	ページのロードが失敗したときに発生します。