Upload for ASP.NET Web Forms

2018.04.17 更新

グレープシティ株式会社

Upload for ASP.NET Web Forms

目次

製品の概要

Upload for ASP.NET Web Forms は、ファイルをアップロードし、サーバーにストリーム送信する簡単で信頼できる方法を提供します。

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

Upload for ASP.NET Web Forms の主な機能のいくつかを以下に示します。

- **複数のファイルのアップロード**
  一度に複数のファイルのアップロードし、一度にアップロードできるファイル数の制限を設定します。
- **大きなファイルのアップロード**
  合計サイズ2 GB までのファイルをアップロードできます。
- **ファイルの自動保管**
  ファイルはまず TempFolder にアップロードされ、次に示す必要な条件が満たされた場合に TargetFolder に移動されます。有効なファイル拡張子、許容された MIME タイプ、許容最大ファイルサイズ、またはカスタム検証ロジックなど。
- **プログレスバーのアップロード**
  Upload for ASP.NET Web Forms は軽量なプログレスバーを備えていますが、クライアント側のアップロードの進捗状況を読み取ることにより、独自の進捗状況 UI を簡単に作成することができます。これにより、現在のアップロード状態についてさまざまな情報を提供します。
- **柔軟なアップロードトリガー**
  Upload for ASP.NET Web Forms には、ファイルをサーバーに送信できる柔軟なトリガーオプションが用意されています。
- **サーバー負荷の低減**
  Upload for ASP.NET Web Forms は、HttpHandler を使用して、クライアントからサーバーに送信されたファイルデータパッケージを読み取ります。ファイルデータはチャンクベースで保存されるため、サーバーのメモリが大量に占有されることはありません。
- **テーマ**
  スマートタグを単にクリックして、6種類のプレミアムテーマ（Arctic、Midnight、Aristo、Rocket、Cobalt、および Sterling）のいずれかを選択して外観を変更します。オプションとして、jQuery UI からThemeRoller を使用してカスタマイズしたテーマを作成します。
- **CSS のサポート**
  CSS（Cascading Style Sheet）のスタイルを使用して、カスタムスキンを定義します。

クイックスタート

このクイックスタートでは、以下の操作を実行できるように **C1Upload** コントロールを使用する方法を学びます。

- **C1Upload**へ複数のファイルを追加する。
- クライアント側のメソッド totalCompleteを使用して、すべてのファイルがアップロードされたときに警告を表示します。
- 最大 100 MB の大規模ファイルを操作可能。

手順 1:プロジェクトの作成とコントロールの追加

このクイックスタートの手順では、ASP.NET Web サイトを作成します。

1. 新しい ASP.NET Web サイトを作成します。
2. 「**デザイン**」タブをクリックして、デザインビューに入ります。
3. Visual Studio ツールボックスで **C1Upload**アイコンをダブルクリックし、コントロールをページに追加します。
   C1Uploadコントロールが web.config ファイルに登録されます。

   > **注意**:プロジェクトがサーバー上にホストされている場合、コントロールを web.config ファイルの <system.web> ノードに手動で登録する必要があります。

   コントロールを手動で登録する必要がある場合は、以下の手順を実行します。
   1. ソースエクスプローラにある Web.config ファイルを開きます。
   2. 以下のマークアップが <system.web> ノードに表示されない場合は、以下のマークアップをファイルに追加します。

      ソースビュー

      ```
      <httpHandlers>
            <add path="C1UploadProgress.axd" verb="*"
      type="C1.Web.Wijmo.Controls.C1Upload.UploadProgressHandler,C1.Web.Wijmo.Controls.4"
      />
          </httpHandlers>
          <httpModules>
            <add name="C1UploadModule"
      type="C1.Web.Wijmo.Controls.C1Upload.UploadModule,C1.Web.Wijmo.Controls.4" />
          </httpModules>
          <compilation debug="true" targetFramework="4.0">
            <assemblies>
              <add assembly="System.Windows.Forms, Version=4.0.0.0, Culture=neutral,
      PublicKeyToken=B77A5C561934E089"/>
              <add assembly="System.Design, Version=4.0.0.0, Culture=neutral,
      PublicKeyToken=B03F5F7F11D50A3A"/>
            </assemblies>
          </compilation>
      ```

   3. 以下の追加のマークアップを <system.webServer> ノードに追加します。

      ソースビュー

      ```
      <system.webServer>
          <validation validateIntegratedModeConfiguration="false"/>
          <modules>
            <add name="C1UploadModule"
      type="C1.Web.Wijmo.Controls.C1Upload.UploadModule,C1.Web.Wijmo.Controls.4" />
          </modules>
          <handlers>
            <add name="C1UploadProgress" path="C1UploadProgress.axd" verb="*"
      type="C1.Web.Wijmo.Controls.C1Upload.UploadProgressHandler,C1.Web.Wijmo.Controls.4"
      />
          </handlers>
      ```

```
</system.webServer>
```

このマークアップは、プロジェクトがサーバー上にホストされているときに必要になります。

4. ページ内の C1Uploadコントロールを選択し、[**プロパティ**]ウィンドウでMaximumFiles プロパティを「5」に設定して、一度に5つのファイルをアップロードできるようにします。

手順 1の完了

この手順では、ASP.NET Web サイトを作成しました。また、**C1Upload**の設定項目を web.config ファイルに追加し、一度にアップロードできるファイルの数を設定しました。次の手順では、ファイルのアップロード完了時に警告ダイアログボックスを表示するために、クライアント側の totalComplete イベントを使用する方法を説明します。

手順 2：アップロードの完了時に警告を表示

この手順では、クライアント側イベント totalCompleteを使用して、すべてのファイルがアップロードされたときに警告ダイアログボックスを表示します。

1. デザインビューに切り替えていない場合は、**[デザイン]** タブをクリックして切り替えます。
2. ページ内の**C1Upload**コントロールを選択し、**[プロパティ]**ウィンドウで**OnClientTotalComplete**プロパティを「totalComplete」に設定します。
3. **[ソース]**タブをクリックして、ソースビューに切り替えます。
4. <asp:Content> タグ内に次のマークアップを追加して、すべてのファイルがアップロードされたときに応答を返す「totalComplete」という名前の JavaScript 関数を割り当てます。

ソースビュー

```
<script type="text/javascript">
function totalComplete(e, data) {
alert("アップロードの完了！");
}
</script>
```

手順2の完了

この手順では、クライアント側イベント totalComplete を使用して、すべてのファイルがアップロードされたときに警告ダイアログボックスを表示しました。次の手順では、アップロード可能なファイルのサイズを決定します。

手順 3：大規模ファイルサイズの有効化

この手順では、web.config ファイルで**maxRequestLength** および**executionTimeout** プロパティの値を設定し、最大 100 MB のファイルを操作可能にします。

1. ソリューションエクスプローラを開いて web.config ファイルを開きます。
2. **MaxRequestLength** と **executionTimeout** の値を、次のように設定します。

ソースビュー

```
<httpRuntime maxRequestLength="102400" executionTimeout="3600" />
```

Web.config ファイルの設定は、次のようになります。

[バージョン7よりも前のIIS]

ソースビュー

```
<configuration>
...
```

```
<system.web>
<httpRuntime maxRequestLength="102400" executionTimeout= "3600" />
...
</system.web>
</configuration>
```

[IIS 7]

ソースビュー

```
<system.webServer>
......
   <security >
     <requestFiltering>
       <requestLimits maxAllowedContentLength="1024000000" />
     </requestFiltering>
   </security>
</system.webServer>
```

この設定では、アップロードできるファイルの最大サイズは 100 MB、アップロード時間は1時間までです。

手順 3の完了

この手順では、最大 100 MB のファイルが操作可能になるように、web.config ファイルで**maxRequestLength**および**executionTimeout**プロパティの値を設定しました。

手順 4:ファイルのアップロード

この最後の手順では、プロジェクトを実行し、**C1Upload** に複数のファイルを追加し、ファイルをアップロードします。

1. **[F5]**を押して、プロジェクトを実行します。
2. **〈ファイルのアップロード** をクリックしてファイルを追加します。
3. ファイルを選択し、〈**Open**〉をクリックします。手順 2と3を繰り返して合計5つのファイルをアップロードします。
4. **〈アップロード〉**  または**〈すべてアップロード〉**  ボタンをクリックしてファイルをアップロードします。[アップロードの完了!]ダイアログボックスが表示されます。

手順 4の完了

この手順では、プロジェクトを実行し、**C1Upload** に複数のファイルを追加し、ファイルをアップロードしました。

デザイン時のサポート

以下のセクションでは、**Upload for ASP.NET Web Forms**で使用可能な各種のサポートについて詳しく説明します。

タスクメニューの起動

Visual Studio では、**Upload for ASP.NET Web Forms**のコントロールは **C1Upload** スマートタグを備えています。スマートタグは、コントロール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の**タスク**メニューを表します。コントロールの右上端のスマートタグ ( ) をクリックして、コントロールの**タスク**メニューを呼び出すことができます。スマートタグの使用方法の詳細については、スマートタグを参照してください。

コンテキストメニューの起動

Upload for ASP.NET Web Forms コンポーネントは、関連付けられたコンテキストメニューを使用して、デザイン時に簡単に設定できます。詳細については、コンテキストメニューを参照してください。

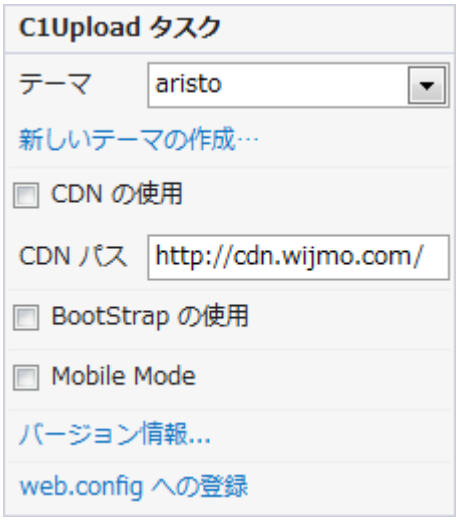
Upload for ASP.NET Web Forms コントロールのプロパティの表示

これらの **Upload for ASP.NET Web Forms's**コントロールのプロパティにアクセスするには、コントロールを右クリックして[**プロパティ**]を選択するか、**プロパティ**ウィンドウのドロップダウンリストボックスからクラスを選択するだけです。

スマートタグ

Visual Studio では、**Upload for ASP.NET Web Forms** の各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、各コントロール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の**タスクメニュー**を表します。

[**C1Upload タスク**]メニューにアクセスするには、**C1Upload**コントロールの右上端にあるスマートタグ をクリックします。これによって、[**C1Upload タスク**]メニューが開きます。



[**C1Upload タスク**]メニューは次のように動作します。

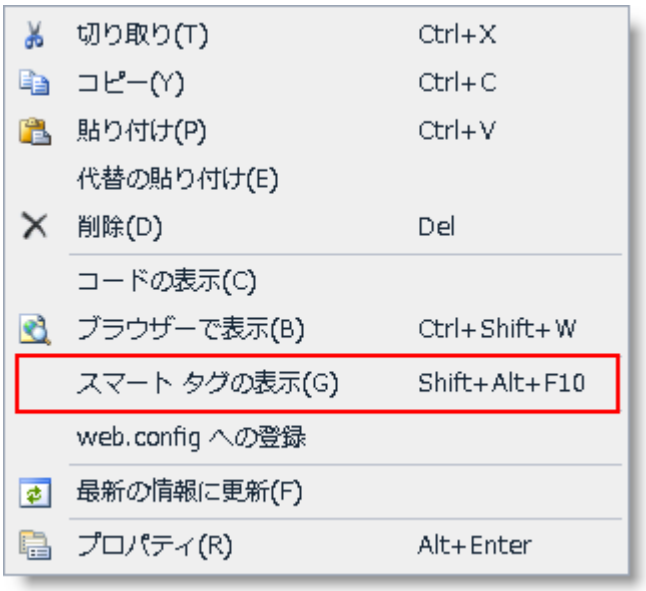
- **テーマ**
  Themeプロパティのドロップダウン矢印をクリックし、任意の組み込みテーマを選択して、コントロールの外観を変更します。
- **新しいテーマの作成**
  [**新しいテーマの作成**]オプションをクリックすると、**ThemeRoller for Visual Studio** が開きます。したがって、開発環境内でテーマをカスタマイズすることができます。アプリケーションで **ThemeRoller for Visual Studio** を使用する方法については、ThemeRoller for Visual Studioを参照してください。
- **CDN の使用**
  [**CDN の使用**]チェックボックスを ON にすると、CDN からクライアントリソースがロードされます。これはデフォルトで

OFF です。

- **CDN パス**
  CDN の URL パスを表示します。
- **Bootstrap の使用**
  [**Bootstrap の使用**]オプションを選択すると、コントロールに Bootstrap テーマを適用することができます。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。
- **バージョン情報**
  [**バージョン情報**]をクリックすると、製品のバージョン情報を確認できるダイアログボックスが表示されます。
- **web.config への登録**
  項目[**Web.config への登録**]をクリックすると、**C1Upload** コントロールの設定項目が Web.config に登録されます。

コンテキストメニュー

Upload for ASP.NET Web Forms C1Upload コントロールは、デザイン時に使用する追加機能のコンテキストメニューを備えています。**C1Upload** コントロールを右クリックして、そのコンテキストメニューを開きます。



C1Upload の外観

6種類のプレミアムテーマ（*Arctic*、*Midnight*、*Aristo*、*Rocket*、*Cobalt*、および *Sterling*）のいずれかを選択して **C1Upload** の外観を変更します。または、jQuery UI から ThemeRoller を使用して独自のカスタマイズしたテーマを作成します。

組み込み Wijmo テーマ

C1Upload for ASP.NET Web Forms には、コントロールを自動的に書式設定できる6種類の組み込みテーマが用意されています。テーマには、**arctic**、**aristo**、**cobalt**、**midnight**、**rocket**、および **sterling** があります。

arctic

以下の画面は **arctic** テーマを表示しています。

aristo

以下の画面は aristo テーマを表示しています。これは、**C1Upload** コントロールのデフォルト書式です。

cobalt

以下の画面は **cobalt** テーマを表示しています。

midnight

以下の画面は **midnight** テーマを表示しています。

rocket

以下の画面は **rocket** テーマを表示しています。

sterling

以下の画面は **sterling** テーマを表示しています。

CSS セレクタ

CSS スタイルを使用して **C1Upload** の要素をスタイル設定し、その外観を真に独特のものにすることができます。カスタマイズ処理を簡素化するために、ComponentOne には、その6種類の組み込みテーマごとに CSS セレクタが組み込まれています。テーマについての詳細は、「 組み込み Wijmo テーマ」を参照してください。

背景、テキスト、フォント、枠、輪郭、マージン、埋め込み、リスト、表などの一般的な CSS プロパティを該当する CSS セレクタに適用できます。

一般に使用される個々の CSS セレクタとグループ化された CSS セレクタのリストについては、プロジェクトの **C1Upload** コントロールを選択し、Visual Studio プロパティウィンドウで **CssClass** プロパティの横にあるドロップダウンリストを表示します。**C1Upload** CSS セレクタは、下図のように wijmo-wijupload から開始します。

wijmo-wijupload
wijmo-wijupload-buttonContainer
wijmo-wijupload-cancel
wijmo-wijupload-file
wijmo-wijupload-loading
wijmo-wijupload-upload

個々の CSS セレクタをグループとして組み合わせ、CSS セレクタをより具体的かつ強力なものにすることができます。

タスク別ヘルプ

タスク別ヘルプのセクションは、Visual Studio ASP.NET 環境でのプログラミングに精通し、**Upload for ASP.NET Web Forms** コントロールを全般的に理解しているユーザーを対象としています。

各トピックでは、**C1Upload** 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示します。各トピックで概説されている手順に従うことによって、さまざまな **C1Upload** 機能を使用したプロジェクトを作成できます。ここでは、Visual Studio で Web プロジェクトまたは Web サイトプロジェクトが作成されていることを前提としています。

C1Upload でのアンチウイルスファイルスキャンの実装

この例では、アップロード可能なファイルとして有効なファイル拡張子を設定し、ファイルをスキャンして、ウイルスに感染していないかどうかを確認します。ファイルがウイルスに感染している場合、アップロードはキャンセルされます。

1. **Label** コントロールを Web プロジェクトに追加します。
2. **C1Upload** コントロールを追加します。
3. **C1Upload** コントロールを選択し、スマートタグをクリックして、[**C1Upload タスク**]メニューで[**web.config への登録**]を選択します。
4. Visual Studio のプロパティウィンドウで、**TargetFolder** プロパティの横に仮想パスと入力します。
5. **ValidFileExtensions** プロパティの横に **.doc,.jpg** と入力します。これは、アップロード可能なファイルのタイプをフィルタリングします。
   ValidatingFile の**C1Upload** イベントがサブスクライブされます。これに対して、AVG アプリケーションのインスタンスを実行し、ファイルが感染しているかどうかをチェックします。この例では、AVG 10(無償版 - http://www.freeavg.com/?lng=in-en&cmpid=free)を使用していることに注意してください。
   ウイルススキャンの結果は Report.txt に書き込まれ、**C1Upload** の一時記憶域に保存されます。
   次に、**StreamReader** オブジェクトを使用して Report.txt ファイルを読み取り、"Found infections" という文字列が含まれているかどうかを確認します。この文字列が含まれている場合は、**e.IsValid = False** を呼び出してアップロードプロセスをキャンセルし、**File.Delete()** を使用してアップロードされたファイルを削除します。

6. [**表示**]→[**コード**]を選択し、次のコードを**Default.aspx.cs** に追加します。

ソースビュー

```
protected void u1_ValidatingFile(object sender, C1.Web.Wijmo.Controls.C1Upload.ValidateFileEventArgs e)
{
var file = e.UploadedFile;
// foreach を削除します
// foreach (C1FileInfo file in C1Upload1.UploadedFiles)
try
{
//ファイルを検証します
// メモ:この引数内の「TempFileName」と「FileName」は読み取り専用プロパティです。
// メッセージは「e.Message」で渡すことができます
}
catch (Exception ex)
{
//例外を処理します
}
}
```

アップロードする画像の表示

この例では、クライアント側イベント「complete」と「upload」を使用して、画像ファイルをアップロードするときに、選択されたファイルを **Image** コントロールに表示します。

1. Web サイトプロジェクトのデザインビューで、ページに **Image** コントロールを追加します。
2. ページに **C1Upload** コントロールを追加し、[プロパティ]ウィンドウで**MaximumFiles** プロパティを **1** に設定します。これで、アップロード時にファイルを1つだけ選択できるようになります。
3. **OnClientComplete** プロパティを **complete** に設定します。
4. **OnClientUpload** プロパティを **upload** に設定します。
5. **[ソース]**タブをクリックして**ソース**ビューに切り替え、次の JavaScript マークアップを**<cc1:C1Upload ... />** マークアップの後に追加します。

ソースビュー

```
<script type="text/javascript">
      var ImageSrc = "";
          function upload(e, args) {
             ImageSrc = "UploadedFiles/" + args[0].files[0].name;
          }
          function complete(e, args) {
             $("#<%=Image1.ClientID %>").attr("src", ImageSrc);
          }
   </script>
```

プロジェクトのマークアップ 全体は次のようになります。

ソースビュー

```
<%@ Page Title="Home Page" Language="C#" MasterPageFile="~/Site.master" AutoEventWireup="true"
   CodeFile="Default.aspx.cs" Inherits="_Default" %>


<%@ Register assembly="C1.Web.Wijmo.Controls.4" namespace="C1.Web.Wijmo.Controls.C1Upload"
TagPrefix="cc1" %>


<asp:Content ID="HeaderContent" runat="server" ContentPlaceHolderID="HeadContent">
</asp:Content>
<asp:Content ID="BodyContent" runat="server" ContentPlaceHolderID="MainContent">


   <%--ページに Image コントロールを追加します。--%>
    <asp:Image ID="Image1" runat="server" />

   <%-- クライアント側イベント「complete」と「upload」を呼び出して、1にアップロードするファイルの最大数を設定します。--%>
    <cc1:C1Upload ID="C1Upload1" runat="server" OnClientComplete="complete"
      OnClientUpload="upload" width="300px" MaximumFiles="1" />


       <script type="text/javascript">
          var ImageSrc = "";


          // アップロードの前に、アップロードの完了時にターゲットフォルダ「UploadedFiles」に保存されるファイルを画像に設定します。
          function upload(e, args) {
             ImageSrc = "UploadedFiles/" + args[0].files[0].name;
          }
```

```
        // アップロードが完了すると、ターゲットフォルダに保存された画像ファイルが Image コントロールに表示されます。
        function complete(e, args) {
            $("#<%=Image1.ClientID %>").attr("src", ImageSrc);
        }
    </script>
</asp:Content>
```

プロジェクトを実行し、**[ファイルのアップロード]**をクリックして、画像ファイルを1つ選択します。**[すべてのファイルをアップロード]**をクリックします。ファイルのアップロードが完了すると、その画像がページの **Image** コントロールに表示されます。

クライアント側の機能

Upload for ASP.NET Web Forms コントロールには、非常に充実したクライアント側オブジェクトモデルがあります。そのメンバは、ほとんどがサーバー側コントロールのメンバと同じです。

C1Upload コントロールが表示されると、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生成されます。これは、サーバーにポストバックしなくても、**C1Upload** コントロールのプロパティやメソッドにアクセスできるということです。

クライアント側コードを使用すれば、時間をかけて Web サーバーに情報を送信しなくても、Web ページに多くの機能を実装できます。そのため、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 サイトの効率を高めることができます。

クライアント側イベント

Upload for ASP.NET Web Forms には、複数のクライアント側イベントが含まれています。それらを利用すれば、ファイルのアップロードなどの処理が行われたときに、**C1Upload** を操作できます。

クライアント側イベントの表にリストされたサーバー側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反応する JavaScript 関数の名前を指定できます。たとえば、「Upload」という JavaScript 関数を割り当てて、ファイルがアップロードされるときに応答させるにはOnClientUpload プロパティを「Upload」に設定します。

下の表に、クライアントスクリプトで使用できるイベントを示します。これらのプロパティはサーバー側で定義されていますが、実際のイベントや各 JavaScript 関数用に宣言する名前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
OnClientChange	change	ユーザーがファイルを選択するときに発生します。
OnClientComplete	complete	ファイルのアップロードが完了したときに発生します。例については、「アップロードする画像の表示」を参照してください。
OnClientProgress	progress	ファイルのアップロードされているときに発生します。
OnClientTotalComplete	totalComplete	〈uploadAll〉ボタンがクリックされ、ファイルのアップロードが完了したときに発生します。
OnClientTotalProgress	totalProgress	〈uploadAll〉ボタンがクリックされ、ファイルがアップロードされているときに発生します。
OnClientTotalUpload	totalUpload	〈uploadAll〉ボタンがクリックされたときに発生します。
OnClientUpload	upload	ファイルがアップロードされる前に発生します。例については、「アップロードする画像の表示」を参照してください。

C1Uploadクライアント側イベントの説明と構文の例については、http://docs.grapecity.com/help/wijmo-3 でも確認できます。