
ComponentOne

Upload for ASP.NET Wijmo

Copyright © 1987-2012 GrapeCity, Inc. All rights reserved.

ComponentOne, a division of GrapeCity

201 South Highland Avenue, Third Floor

Pittsburgh, PA 15206 • USA

Internet: info@ComponentOne.com

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

Sales

E-mail: sales@componentone.com

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

Trademarks

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

Warranty

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

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

Copying and Distribution

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

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

Table of Contents

ComponentOne Upload for ASP.NET Wijmo Overview	1
Help with ComponentOne Studio for ASP.NET Wijmo	1
Key Features	1
Wijmo Top Tips	2
Upload for ASP.NET Wijmo Quick Start	3
Step 1 of 4: Create the Project and add C1Upload	3
Step 2 of 4: Set Up Target and Temp Folders	3
Step 3 of 4: Enable Large File Size	3
Step 4 of 4: Upload the Files	4
Upload for ASP.NET Wijmo Client-Side Reference	5
Using the Wijmo CDN	5
Design-Time Support	7
Upload for ASP.NET Wijmo Smart Tag	7
Upload for ASP.NET Wijmo Context Menu	8
Upload for ASP.NET Wijmo Appearance	8
Built-in Wijmo Themes	8
ThemeRoller Overview	9
C1Upload CSS Selectors	10
Upload for ASP.NET Wijmo Task-Based Help	11
Implementing an Anti-Virus File Scan in C1Upload	11
Working with the Client-Side	13
Client-Side Events	13

ComponentOne Upload for ASP.NET Wijmo Overview

ComponentOne Upload™ for ASP.NET Wijmo provides a simple and reliable way to upload files and streams to the server.

Help with ComponentOne Studio for ASP.NET Wijmo

Getting Started

For information on installing ComponentOne Studio for ASP.NET Wijmo, licensing, technical support, namespaces and creating a project with the control, please visit [Getting Started with Studio for ASP.NET Wijmo](#).

What's New

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

Key Features

The following are some of the main features of **Upload for ASP.NET Wijmo** that you may find useful:

- **Multi-file Upload**
Upload multiple files at once, and set restrictions on how many files can be uploaded at once.
- **Upload Large Files**
You can upload files with a combined size of up to 2GB.
- **Automatic File Storing**
Files are first uploaded to TempFolder and will be moved to the TargetFolder if they met the required conditions. For example, valid file extensions, the allowed MIME type, the allowed maximum file size, or custom validating logic.
- **Upload Progress Bar**
Upload for ASP.NET Wijmo provides a light-weight progress bar, but you can easily provide your own progress UI by reading the client-side upload progress, which provides rich information for current uploading states.
- **Flexible Upload Triggers**
Upload for ASP.NET Wijmo provides flexible trigger options that allow you to control when to submit the files to server.
- **Reduces Server Load**
Upload for ASP.NET Wijmo uses HttpHandler to read the file data package transmitted from client to server. File data is saved based on chunks, which do not occupy much server memory.
- **Theming**

With just a click of the SmartTag, change the button's look by selecting one of the 6 premium themes (Arctic, Midnight, Aristo, Rocket, Cobalt, and Sterling). Optionally, use ThemeRoller from jQuery UI to create a customized theme!

- **CSS Support**

Use a cascading style sheet (CSS) style to define custom skins. CSS support allows you to match the upload control to your organization's standards.

Wijmo Top Tips

The following tips may help you troubleshoot when working with Studio for ASP.NET Wijmo.

Tip 1: Prevent poor page rendering in quirks mode by editing the meta tag to fix rendering.

If a user's browser is rendering a page in quirks mode, widgets and controls may not appear correctly on the page. This is indicated by a broken page icon in the address bar. In **Compatibility View**, the browser uses an older rendering engine.



Users can set this view that causes the issue. To prevent rendering in quirks mode, you can force the page to render with the latest browser. Add the following meta tag to the header of the page:

```
<meta http-equiv="X-UA-Compatible" content="IE=edge,chrome=1" />
```

Upload for ASP.NET Wijmo Quick Start

In this quick start you will learn how to use the C1Upload control so you can do the following:

- Use the TargetFolder and TempFolder properties for automatic storing.
- Add multiple files to C1Upload.
- Enable large file sizes up to 100MB.
- Upload the files to **Target** and **Temp** folders.

Step 1 of 4: Create the Project and add C1Upload

In this step of the quick start, you will create your ASP.NET Web site.

1. Create a new ASP.NET Web site.
2. Click the **Design** tab to enter Design view.
3. In the Visual Studio Toolbox, double-click the **C1Upload** icon to add the control to your page.
4. Click the smart tag to open **C1Upload**'s task menu and click **Register in web.config** to register the **C1Upload** control in your web.config file.

Step 1 of 4 Completed

In this step, you created an ASP.NET Web site. You also added the configuration items for C1Upload to the web.config file. In the next step, you will set up the target and temp folders for the uploaded files to be stored.

Step 2 of 4: Set Up Target and Temp Folders

In this step you will create two folders on your drive to store the temporary and valid uploaded files. You will also change the default value of the MaximumFiles property to 5 to allow up to 5 files to be added to C1Upload.

1. Create a folder (C:\UploadFolder) on disk [C:].
2. Create two sub folders under that folder, one with name, **Temp**, the other with name **Target** as below:
C:\UploadFolder\Temp
C:\UploadFolder\Target
3. In your project open the .aspx file and change the value of the TempFolder property to "C:\UploadFolder\Temp" and change the value of the TargetFolder property to "C:\UploadFolder\Target" in the Visual Studio Properties window.
4. Set the MaximumFiles property to "5" to enable 5 files to add to **C1Upload**.

Step 2 of 4 Completed

In this step, you set up the folders on your drive for the uploaded files, assigned the file paths to TempFolder and TargetFolder properties, and set the MaximumFiles property to 5 to enable 5 files to be added at run time.

Step 3 of 4: Enable Large File Size

In this step you will set the value of the **maxRequestLength** and **executionTimeout** properties in your web.config file to enable file sizes up to 100MB.

1. Open the Solution Explorer, navigate to the web.config file and open it.

2. Set the **MaxRequestLength** and **executionTimeout** values to the following:

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

The settings for the Web.config file should appear like the following:

[IIS prior to version 7]

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

[IIS 7]

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

This configuration enables uploading of files up to 100MB and upload periods up to 1 hour.



Step 3 of 4 Completed

In this step you set the value of the **maxRequestLength** and **executionTimeout** properties in your web.config file to enable file sizes up to 100MB.

Step 4 of 4: Upload the Files

In this last step you will run your project, add multiple files to C1Upload, and upload the files to the **Temp** and **Target** folders you have created in step 1.

1. Press **F5** to run your project.
2. Click **Upload files** to add a file.
3. Select a file and click **Open**. You can repeat step 2 and step 3 to upload a total of five files.

4. Click the **Upload**  or **Upload All**  button to upload the file(s).
5. Navigate to the **Temp** folder on your drive and notice the file you uploaded exists in the **Temp** folder.
6. Navigate to the **Target** folder on your drive and notice the file you uploaded exists in the **Target** folder because it met the requirements.

Step 4 of 4 Completed

In this step you ran your project, added multiple files to **C1Upload**, and uploaded the files to the **Temp** and **Target** folders.

Upload for ASP.NET Wijmo Client-Side Reference

As part of the amazing [ComponentOne Web stack](#), the Wijmo jQuery UI widgets are optimized for client-side Web development and utilize the power of jQuery for superior performance and ease of use.

The ComponentOne Wijmo website at <http://wijmo.com/widgets/> provides everything you need to know about Wijmo widgets, including demos and samples, documentation, theming examples, support and more.

The client-side documentation provides an overview, sample markup, options, events, and methods for each widget. To get started with client-side Web development for **Upload for ASP.NET Wijmo**, view our [Wijmo wiki documentation](#). Note that the **Overview** topic for each of the widgets applies mainly to the widget, not to the server-side ASP.NET Wijmo control.

- [Uploader Options](#)
- [Uploader Events](#)
- [Uploader Methods](#)

Using the Wijmo CDN

You can easily load the client-side Wijmo widgets into your web page using a Content Delivery Network (CDN). CDN makes it quick and easy to use external libraries, and deploy them to your users. A CDN is a network of computers around the world that host content. Ideally, if you're in the United States and you access a webpage using a CDN, you'll get your content from a server based in the US. If you're in India or China, and you access the SAME webpage, the content will come from a server a little closer to your location.

When web browsers load content, they commonly will check to see if they already have a copy of the file cached. By using a CDN, you can benefit from this. If a user had previously visited a site using the same CDN, they will already have a cached version of the files on their machine. Your page will load quicker since it doesn't need to re-download your support content.

Wijmo has had CDN support from the very beginning. You can find the CDN page at <http://wijmo.com/downloads/cdn/>. The markup required for loading Wijmo into your page looks similar to this:

```
<!--jQuery References-->
<script src="http://ajax.aspnetcdn.com/ajax/jquery/jquery-1.7.1.min.js"
type="text/javascript"></script>
<script src="http://ajax.aspnetcdn.com/ajax/jquery.ui/1.8.17/jquery-
ui.min.js" type="text/javascript"></script>
<!--Theme-->
<link href="http://cdn.wijmo.com/themes/rocket/jquery-wijmo.css"
rel="stylesheet" type="text/css" title="rocket-jqueryui" />
<!--Wijmo Widgets CSS-->
<link href="http://cdn.wijmo.com/jquery.wijmo-complete.all.2.0.0.min.css"
rel="stylesheet" type="text/css" />
<!--Wijmo Widgets JavaScript-->
<script src="http://cdn.wijmo.com/jquery.wijmo-open.all.2.0.0.min.js"
type="text/javascript"></script>
<script src="http://cdn.wijmo.com/jquery.wijmo-complete.all.2.0.0.min.js"
type="text/javascript"></script>
```

In this markup, you'll notice that some of the .js files are labeled as *.min.js. These files have been minified - in other words, all unnecessary characters have been removed - to make the pages load faster. You will also notice that there are no references to individual .js files. The JavaScript for all widgets, CSS, and jQuery references have been combined into one file, respectively, such as wijmo-complete.2.0.0.min.js. If you want to link to individual .js files, see the **Dependencies** topic for each widget.

With the **ComponentOne Studio for ASP.NET Wijmo** controls, you can click the **Use CDN** checkbox in the control's **Tasks** menu and specify the **CDN path** if you want to access the client-side widgets.



Design-Time Support

The following section details each type of support available in **Upload for ASP.NET Wijmo**.

Invoking the Tasks Menu

In Visual Studio, the **C1Upload** control in **Upload for ASP.NET Wijmo** includes a smart tag. A smart tag represents a short-cut **Tasks** menu that provides the most commonly used properties in a control. You can invoke a control's **Tasks** menu by clicking on the smart tag (🔗) in the upper-right corner of the control. For more information on how to use the smart tag in **Upload for ASP.NET Wijmo**, see [Upload for ASP.NET Wijmo Smart Tag](#) (page 7).

Invoking the Context Menu

You can easily configure the **Upload for ASP.NET Wijmo** component at design time by using its associated context menu. For more information, see the [Upload for ASP.NET Wijmo Context Menu](#) (page 8).

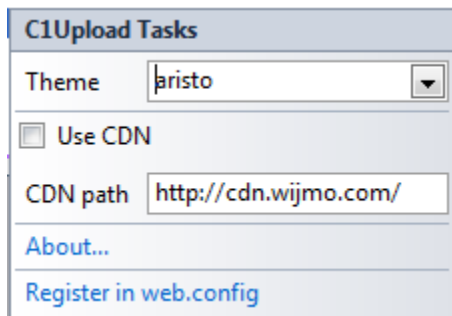
Showing the Upload for ASP.NET Wijmo Control's Properties

You can access the properties for any of **Upload for ASP.NET Wijmo's** controls simply by right-clicking on the control and selecting **Properties** or by selecting the class from the drop-down list box of the **Properties** window.

Upload for ASP.NET Wijmo Smart Tag

In Visual Studio, each control in **Upload for ASP.NET Wijmo** includes a smart tag (🔗). A smart tag represents a short-cut **Tasks** menu that provides the most commonly used properties in each control.

To access the **C1Upload Tasks** menu, click the smart tag (🔗) in the upper-right corner of the C1Upload control. This will open the **C1Upload Tasks** menu.



The **C1Upload Tasks** menu operates as follows:

- **Theme**
Click the drop-down arrow in the **Theme** property to select one of the built-in themes to change the appearance of the control.
- **Use CDN**
Check the **Use CDN** check box to link to the content delivery network to access Wijmo widgets and cascading style sheets.
- **CDN path**

The CDN path points to the content delivery network where you can access Wijmo widgets and cascading style sheets.

- **About**

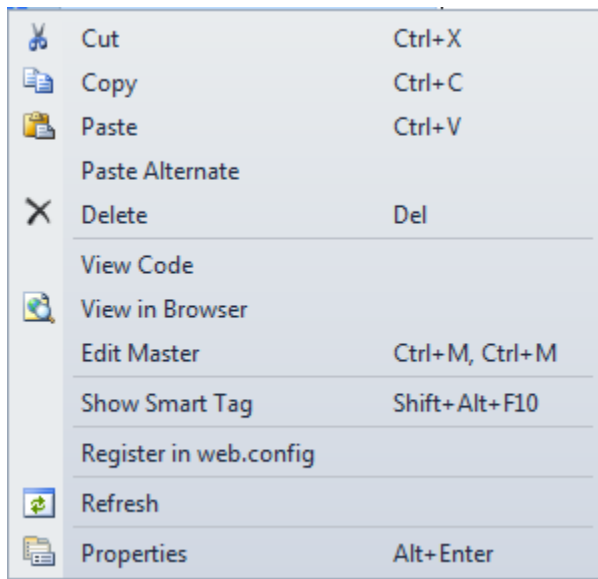
Clicking the **About** item displays the **About ComponentOne Studio for ASP.NET Wijmo** dialog box, which is helpful in finding the version number of **Studio for ASP.NET Wijmo** and online resources.

- **Register in web.config**

Clicking on the **Register in Web.config** item registers the configuration items in the Web.config for the **C1Upload** control.

Upload for ASP.NET Wijmo Context Menu

The **Upload for ASP.NET Wijmo C1Upload** control provides a context menu for additional functionality to use at design time. Right-click the **C1Upload** control to open the following context menu:



Upload for ASP.NET Wijmo Appearance


Change the **C1Upload** control's look by selecting one of the six premium themes (*Arctic*, *Midnight*, *Aristo*, *Rocket*, *Cobalt*, and *Sterling*). Or you can use ThemeRoller from jQuery UI to create your own customized theme!

Built-in Wijmo Themes

C1Upload for ASP.NET Wijmo provides six built-in themes to automatically format the control. The themes include the following: **arctic**, **aristo**, **cobalt**, **midnight**, **rocket**, and **sterling**.


arctic

The following image displays the **arctic** theme:

A rectangular button with a light gray background and rounded corners. The text "Upload files" is centered in a blue, sans-serif font.

aristo

The following image displays the **aristo** theme. This is the default format for the **CIUpload** control:

A rectangular button with a light gray background and rounded corners. The text "Upload files" is centered in a blue, sans-serif font.


cobalt

The following image displays the **cobalt** theme:

A rectangular button with a light blue background and rounded corners. The text "Upload files" is centered in a white, sans-serif font.

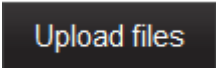
midnight

The following image displays the **midnight** theme:

A rectangular button with a dark blue background and rounded corners. The text "Upload files" is centered in a white, sans-serif font.

rocket

The following image displays the **rocket** theme:

A rectangular button with a dark gray background and rounded corners. The text "Upload files" is centered in a white, sans-serif font.

sterling

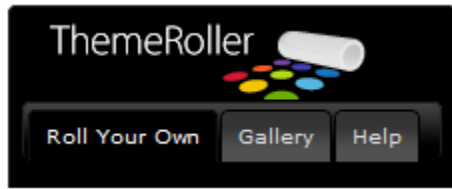
The following image displays the **sterling** theme:

A rectangular button with a light gray background and rounded corners. The text "Upload files" is centered in a blue, sans-serif font.

ThemeRoller Overview

One of the most convenient features of Wijmo is its ThemeRoller compatibility. ThemeRoller is a Web application for creating unique jQueryUI themes to skin your Web app's widgets. With its simple interface and WYSIWYG preview, you can create a skin for all of your Wijmo widgets and other ThemeRoller-compatible widgets in less time than it would take to open a graphic editor.

We can find the ThemeRoller Web app at <http://jqueryui.com/themeroller/>. Once you've arrived at the app, take a gander at the column on the left-hand side of the page and observe that it holds three tabs: **Roll Your Own**, **Gallery**, and **Help**.



The Roll Your Own Tab

Roll Your Own is where the magic happens. From this tab, we have the power to tweak your theme to perfection. To change an element, all we have to do is expand a node and get to work.

The Gallery Tab

Clicking the **Gallery** contains a list and preview of ThemeRoller's premium themes. From here, we can preview, download, or edit one of the ready-made themes in the following ways:

- Clicking the snippet view of the theme loads an interactive preview to the right of the gallery.
- Clicking a theme's **Download** button will take us to the **Build Your Download** page. If we want to download one of the premade themes to skin your Wijmo widgets, all we have to do is navigate to the light orange panel, select our **Advanced Theme Settings** and **Version**, and click **Download**.
- Clicking **Edit** will load the chosen theme and return us to the **Roll Your Own** tab, where we can use the ThemeRoller to tweak the theme.

The Help Tab

Clicking the **Help** tab will give us a quick reference and special information regarding the ThemeRoller, such as information for plugin developers and browser support notices.

For more information on using the ThemeRoller and for a short tutorial, visit our [Wijmo website](#).

C1Upload CSS Selectors

You can style any **C1Upload** elements using CSS styles to make their appearance truly unique. To make the customization process easier, ComponentOne includes CSS selectors for each of its six built-in themes. For more information on themes, see [Built-in Wijmo Themes](#) (page 8).

You can apply general CSS properties such as background, text, font, border, outline, margin, padding, list, and table to applicable CSS selectors.

For a list of common individual CSS selectors and grouped CSS selectors, select the C1Upload control in your project, and view the drop-down list next to the **CssClass** property in the Visual Studio Properties window.

C1Upload CSS selectors start with wijmo-wijupload:

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

You can combine the individual CSS selectors as a group to make the CSS selector more specific and strong.

Upload for ASP.NET Wijmo Task-Based Help

The task-based help section assumes that you are familiar with programming in the Visual Studio ASP.NET environment and have a general understanding of the **Upload for ASP.NET Wijmo** control.

Each topic provides a solution for specific tasks using the C1Upload control. By following the steps outlined in each topic, you will be able to create projects using a variety of C1Upload features.

Implementing an Anti-Virus File Scan in C1Upload

1. Create a folder on your local **C:** drive called **Upload (C:\Upload)**.
2. Within the Upload folder, create a **Temp** folder and **Target** folder (C:\Upload\Temp, C:\Upload\Target).
3. Add a **Label** control to your Web project.
4. Add a C1Upload control to your Web project.
5. Select the C1Upload control, click the smart tag, and select **Register in web.config** in the **C1Upload Tasks** menu.
6. In the Visual Studio Properties window, enter **C:\Upload\Target** next to the TargetFolder property.
7. Enter **C:\Upload\Temp** next to the **C1Upload.Temp** property.
8. Next to the ValidFileExtensions property, enter **.doc, .jpg**. This will filter the type of files that can be uploaded.

We will subscribe the ValidatingFile event of C1Upload and scan the file being uploaded. For this, we will run an instance of the AVG application and check whether the file is infected or not. Note that for this example we are using AVG 10 (free version - <http://www.freeavg.com/?lng=in-en&cmpid=free>).

The results of the virus scan are written in a Report.txt which is saved in the Temporary storage of C1Upload (**C:\Upload\Temp** folder).

Next, we will use a **StreamReader** object to read the Report.txt file and check whether it contains a "Found infections" string. If the string is found, then we will cancel the upload process by calling **e.IsValid = False** and we will then delete the file being uploaded using **File.Delete()**.

9. Select **View | Code** and add the following code to the **Default.aspx.cs**:

```
protected void C1Upload1_ValidatingFile(object sender,
C1.Web.Wijmo.Controls.C1Upload.ValidateEventArgs e)
{
    foreach (C1FileInfo file in C1Upload1.UploadedFiles)
    {
        try
        {
            //do av check here
            Process myProcess = new Process();

            //address of command line virus scan exe
            myProcess.StartInfo.FileName = "C:\\Program
Files\\AVG\\AVG10\\avgscana.exe";
            string myprocarg = "/SCAN=" ;
```

```

        e.UploadedFile.TempFileName("
/REPORT=C:\\Upload\\Temp\\Report.txt");
        myProcess.StartInfo.Arguments = myprocarg;
        myProcess.Start();
        myProcess.WaitForExit(); //wait for the scan to complete

        //add some time for report to be written to file
        int j = 0;
        int y = 0;
        for (j = 0; j <= 1000000; j )
        {
            y = y + 1;
        }

        //Get a StreamReader class that can be used to read the
file
        StreamReader objStreamReader = default(StreamReader);
        objStreamReader =
File.OpenText("C:\\Upload\\Temp\\Report.txt");
        if (!objStreamReader.ReadToEnd().Contains("Found infections
: 0"))
        {
            e.IsValid = false;
            File.Delete(e.UploadedFile.TempFileName);
        }
        objStreamReader.Close();

        if (e.IsValid)
        {
            Labell1.Text = "Your File Uploaded Sucessfully at server
as: ";
            e.UploadedFile.FileName("It was checked with AVG Free
Anti-Virus.");
        }
        else
        {
            Labell1.Text = "Error! ";
            e.UploadedFile.FileName(" contains a virus. It has been
deleted from the server.");
        }
    }
    catch (Exception Exp)
    {
        Labell1.Text = "An Error ocured & all files have been
removed. Please check the attached file(s).";
        e.IsValid = false;
        File.Delete(e.UploadedFile.TempFileName);
    }
}
}

```


Working with the Client-Side

The **Upload for ASP.NET Wijmo** controls have a very rich client-side object model since most of their members are identical to the members in the server-side control.

When a **CIUpload** control is rendered, an instance of the client-side control will be created automatically. This means that you can enjoy the convenience of accessing properties and methods of the **CIUpload** controls without having to postback to the server.

Using client-side code, you can implement many features in your Web page without the need to send information to the Web server, which takes time. Thus, using the client-side object model can increase the efficiency of your Web site.

Client-Side Events

Upload for ASP.NET Wijmo includes several client-side events that allow you to manipulate **CIUpload** when an action occurs such as a file being uploaded.

You can use the sever-side properties, listed in the Client Side Event table, to specify the name of the JavaScript function that will respond to a particular client-side event. For example, to assign a JavaScript function called "Upload" to respond when file is being uploaded, you would set the `OnClientUpload` property to "Upload".

The following table lists the events that you can use in your client scripts. These properties are defined on the server side, but the actual events or the name you declare for each JavaScript function are defined on the client side.

Event Server-Side Property Name	Event Name	Description
<code>OnClientChange</code>	change	Occurs when the user selects a file.
<code>OnClientComplete</code>	complete	Occurs when the file upload is complete.
<code>OnClientProgress</code>	progress	Occurs when the files is uploading.
<code>OnClientTotalComplete</code>	totalComplete	Occurs when the uploadAll button is clicked and the file upload is complete.
<code>OnClientTotalProgress</code>	totalProgress	Occurs when the uploadAll button is clicked and the file is uploading.
<code>OnClientTotalUpload</code>	totalUpload	Occurs when the uploadAll button is clicked.
<code>OnClientUpload</code>	upload	Occurs before the file is uploaded.

Descriptions and syntax examples for the **CIUpload** client-side events can be found at <http://wijmo.com/wiki/index.php/Upload>.