ASP.NET Upload Script

ASP.NET allows for the easy creation of upload handler scripts. The following ASP.NET page allows files to be uploaded. The *fileupload.aspx* page does not process the upload; this task is performed by the *fileupload.aspx.cs* page.

> The source for EditLive multimedia upload scripts can also be found in the SDK_INSTALL/webfolder/uploadscripts/ directory where SDK_INSTALL represents the directory to where the EditLive! SDK is installed.

Defining the Location of the Image Upload Handler Script

The location of the image upload handler script must be defined within the EditLive! configuration file. This setting is configured via the **href** attribute of the <httpUpload> element of the configuration file. To use this example script, the **href** attribute should point to the location of this script on the server.

Defining the Location of the Image Upload Directory

This example script uploads images to the directory specified by the *imageDir* variable. In order for images to function correctly within EditLive!, the base attribute of the <httpUpload> element must reflect the location of the directory where images are to be stored on the Web server.

Example Image Upload Handler Script

Tiny has written a sample image upload handler script using Active Server Pages and VBScript. This script can be found at *SDK_INSTALL\webfolder\uploadscripts\aspnet\fileUpload.aspx* and *SDK_INSTALL\webfolder\uploadscripts\aspnet\fileUpload.aspx.cs*, where SDK_INSTALL represents the location where the EditLive! SDK is installed.

ASP.NET allows for the easy creation of upload handler scripts. The following ASP.NET page allows files to be uploaded. The *fileupload.aspx* page does not process the upload; this task is performed by the *fileupload.aspx.cs* page. The code for the *fileupload.aspx* page is as follows:

```
<%@ Page language="c#" Codebehind="fileupload.aspx.cs" AutoEventWireup="false" Inherits="Ephox.FileUpload" %>
```

The POST is then handled in the **Page_Load** method of the *fileupload.aspx.cs* page. This appears as follows:

```
private void Page_Load(object sender, System.EventArgs e)
{
   /*
    * Set the "path" variable to the location where images are to be stored
    */
   string path = "../images";

   HttpFileCollection files;
   files = Page.Request.Files;

   for(int index=0; index < files.AllKeys.Length; index++)
   {
      HttpPostedFile postedFile = files[index];
      string fileName = null;
      int lastPos = postedFile.FileName.LastIndexOf('\\');
      fileName = postedFile.FileName.Substring(++lastPos);

      //Check the file type through the extension
      if(fileName.EndsWith("jpg") || fileName.EndsWith("jpeg") ||
         fileName.EndsWith("gif") || fileName.EndsWith("png") ||
         fileName.EndsWith("tiff"))
      {
         postedFile.SaveAs(MapPath(path + "/" + fileName));
      }
   }
}
```

Configuring EditLive! to use the Image Upload Script

Below are the steps required to use the ASP.NET image upload handler from above with EditLive! in your own Web application.

1. One line of code in the fileUpload.aspx.cs file must be changed for image upload.

- This line of code specifies the location where you wish image files to be uploaded to. If the location of the upload acceptor script was *http://www.yourserver.com/scripts/fileUpload.aspx*, then setting the path variable to *../images* would upload the images to a directory with the URL *http://www.yourserver.com/images/*.

```
string path = "../images";
```

> Relative paths specified within the image upload acceptor script are relative to the Web accessible location of the image upload acceptor script.

2. EditLive!'s configuration file should now be edited to reflect the changes made in the previous step. You will find these settings within the <httpUpload> element. The URL setting should reflect the location of the *fileUpload.aspx*file on your Web server.
   - The following example reflects the setting of the **href** attribute of the <httpUpload> element if the upload script was at the URL *http://www.yourserver.com/scripts/fileUpload.aspx*.

```
<editLive>
   ...
   <mediaSettings>
      <httpUpload
         base= ...
         href="http://www.yourserver.com/scripts/fileUpload.aspx" />
      ...
   </mediaSettings>
   ...
</editLive>
```

3. Finally the HTTP Image Upload **base**attribute should be changed to reflect the location where images can be found on your Web server.

> This location may not be the same value as that used within the upload acceptor script above. Rather, it will be the virtual directory alias used by your Web server for the location listed in the upload acceptor script.

   - This example follows from the code above. It uses an absolute URL as the value of the base attribute. The value of the base attribute corresponds to the URL that for the directory that images are uploaded to.

```
<editLive>
   ...
   <mediaSettings>
      <httpImageUpload
         base="http://www.yourserver.com/images/"
         href="http://www.yourserver.com/scripts/fileUpload.aspx" />
      ...
   </mediaSettings>
   ...
</editLive>
```

Extending the Image Upload Script for Use with Other File Types

The ASP.NET upload acceptor script provided with EditLive! can, by default, only be used with common image file types. This is restricted by inspecting the extension of an uploaded file. Support for other file types, including multimedia object types such as Macromedia Flash (.swf) and Apple QuickTime (.mov), can easily be added by adding the relevant extension to the list of allowed extensions in the upload *fileupload.aspx.cs* portion of the acceptor script.

The list of permitted extensions can be found as follows in the ASP.NET *fileupload.aspx.cs* upload acceptor script file in the following statement:

```
HttpFileCollection files;
files = Page.Request.Files;

for(int index=0; index < files.AllKeys.Length; index++)
{
   HttpPostedFile postedFile = files[index];
   string fileName = null;
   int lastPos = postedFile.FileName.LastIndexOf('\\');
   fileName = postedFile.FileName.Substring(++lastPos);

   //Check the file type through the extension
   if(fileName.EndsWith("jpg") || fileName.EndsWith("jpeg") ||
      fileName.EndsWith("gif") || fileName.EndsWith("png") ||
      fileName.EndsWith("tiff"))
   {
      postedFile.SaveAs(MapPath(path + "/" + fileName));
   }
}
```

The example below demonstrates how this example can be extended to handle Flash and QuickTime files. Note the addition of the file extensions to the *if* statement in the code below:

```
HttpFileCollection files;
files = Page.Request.Files;

for(int index=0; index < files.AllKeys.Length; index++)
{
   HttpPostedFile postedFile = files[index];
   string fileName = null;
   int lastPos = postedFile.FileName.LastIndexOf('\\');
   fileName = postedFile.FileName.Substring(++lastPos);

   //Check the file type through the extension
   if(fileName.EndsWith("jpg") || fileName.EndsWith("jpeg") ||
      fileName.EndsWith("gif") || fileName.EndsWith("png") ||
      fileName.EndsWith("tiff") || fileName.EndsWith("swf") ||
      fileName.EndsWith("mov"))
   {
      postedFile.SaveAs(MapPath(path + "/" + fileName));
   }
}
```

See Also

- HTTP Upload Support for Images and Objects